

# The microbee Computer-in-a-Book represents a novel, practical effective low cost way of advancing to disk drive capability



The microbee 3.5 inch 400K high speed/high density Drives are very compact, extremely tough and represent the best value-formoney 'Real Disk Drive' on the market today. The Computer-ina-Book comes complete with all the software you will need to get started including demo program, MW Basic, Wordbee Wordprocessing, Telcom communications and CP/M with utilities.

Microbee 64K with Single Disk Drive **\$995** incl. Tax Add-on Disk Drive in matching volume \$345 incl. Tax Ask about Modems, Printers and Business Software.

PHONE ORDERS | bankcard ACCEPTED

Designed and manufactured in Australia by

#### microbee technology centres

1 Pattison Ave, Waitara 2077 Phone (02) 487 2711

729 Glenferrie Rd, Hawthorn 3122 Phone (03) 819 5288

141 Stirling Highway, Nedlands Phone (09) 386 8289

151 Unley Rd, Unley 5061, Phone (08) 272 1384

455 Logan Rd, Stones Corner, 4120 Phone (07) 394 3688

**FACTORY** 

Koala Crescent, West Gosford 2250 Phone (043) 24 2711

# LEDGERMASTER Son

BY LINDSAY R. FORD Dreamcards Software

To start our 32-page section of bumper-tobumper pocket programs, we've purchased the rights to print details of this commercial ackage written by Lindsay Ford. It'll take a bit of typing ...

WITH TAXATION time almost here again, many readers will be faced with the daunting prospect of having to shuffle through bundles of old receipts and cheque butts to put together their tax returns. This problem is a particular burden to people like me, who run small businesses that don't make enough to justify paying the fees of a good tax accountant. Club treasurers facing the task of preparing their annual reports will also know just what I mean!

One solution is to invest in a software accounts package, but these are generally disk-based, expensive to buy and more sophisticated than most of us need. LedgerMaster was written to overcome these problems. It is a financial database program designed to run on 32 Kbyte (or larger) Microbee computers and has all the features you're likely to need to prepare a profit and loss statement for taxation purposes or for publication in your club magazine. It allows you to create a ledger containing up to 320 transaction records (files'), which can be stored on tape, printed, sorted and searched as with any general database.

Where LedgerMaster stands out, however, is it also conducts a balance of the ledger files (and of any other ledgers you have 'chained' to it) and compiles these into a proper profit and loss statement. This takes almost

the effort out of preparing tax returns or treasurers' reports, whether , ou're typing in the receipts and payments in one frenzied, last-minute burst or simply tallying a ledger you've created through the year as transactions occurred.

#### **Program Entry**

The program is really a compromise between efficient code and legibility. Ideally, a BASIC program needs as many instructions as possible crammed into each line, if maximum speed and economic memory use are to be achieved, but this would have made LedgerMaster extremely difficult to read. I've opted for slightly less-than-optimum line lengths, but even so the size of the program means you have to be extremely careful as you type it in. Please bear the following points in mind:

■ Use AUTO mode to enter the program, as this minimises the risk of 'non-printing characters' being accidentally included in the code. If you have my BASIC Screen Editor fitted to your Bee, use it. This will avoid the problem entirely.

- REMs that appear in a line of code (for example in line 1) should be left out entirely.
- Where a line contains a REM but no BASIC code (for example lines 19 and 20), the REM should be entered, but any text following it omitted.
- The up-arrow character (↑) appears frequently in the program. Don't leave it out.
- Watch out for variables I, O and O. These are easy to confuse with ones and zeros.

Once the program has been entered exactly as shown in the listing

(subject to the above comments), use the GX command to locate all lines containing REMs. These lines should be deleted (using the DELETE command) and several copies of the program should then be stored on tape.

At this stage you have an almost complete program. Now type in the following (in 'direct mode' – don't give it a line number):

CLS: X=0: Y=PEEK(2258) + 256\*PEEK(2259): FOR Z=2304 TO Y: X=X+PEEK(Z): NEXT Z: PRINT "X="; X, "Y="; Y: END

When you press the <RETURN> key the screen will clear, and after a few minutes two numbers will appear. 'Y' is the address in decimal of the top of the BASIC file and should be 11819. If it's higher than this you've put something in the program that shouldn't be there, if lower you've left something out. 'X' is a checksum — it is the total of the numbers contained in each of the memory bytes making up the program. If it is 16146 then all is well and you can proceed to the next stage; otherwise you'll have to look for your typing error.

Stage 2 of the operation involves removal of unnecessary spaces in the listing, as these greatly reduce execution speed. Enter "GX///"<Return> and rest a brick on the full-stop key, then go and make yourself a cup of coffee. After about five minutes all spaces will have been removed. Now you'll have to insert spaces in the places they're meant to be by substituting a space for each up-arrow. Use "GX/↑/"to do this. Once done you should save a few copies on tape, then enter the test routine shown above. This time Y should be 10079 and X, 17639. If they are you've got a fully operational LedgerMaster.

#### **Using the Program**

When you RUN LedgerMaster there will be a brief delay as the main arrays are initialised, then a menu will appear giving you the option to 'Create Base' or 'Load'. You will also see two numbers at the bottom right of the screen. The first of these is the free memory counter, which tells you how much room remains in the ledger (you can't run out of file space on a 32 Kbyte machine), and the second is the number of free files available. As the ledger hasn't been initialised this second number will be zero. You won't have a ledger tape you can load at this stage, so let's look at the first option.

#### **Create Base**

This routine initialises the ledger, clearing out any existing material in memory. For this reason you will be asked if you are 'Sure?' if you access it when a ledger is already present. You will then be requested to name the ledger (maximum 24 characters), then to input the various categories of payments and receipts you want included on the final balance sheet (such as 'insurance', 'rent', 'royalties', 'wages' and so on). Give these a good deal of thought before starting, as they comprise the framework around which your financial report or tax return will be built.

You can enter up to 20 classes of payments and 20 of receipts (maximum 12 characters), and a 'carried forward' balance for each. This balance will be of importance if you are including amounts accrued from a previous period, such as debts outstanding from the last tax year. No 'carried forward' balance will be entered if you press the <Return> key. Pressing <Return> when you're prompted to input a payment or receipt category indicates you've finished entering items in that class.

Once the payments and receipts categories are entered your ledger 'base' is complete and the program will return to the menu. You will now have four options available — 'Create', 'Load', 'Enter' and 'Save'. If you >

# Microbee 🐷

want to go on and input a couple of files, the first option to look at is 'Enter'.

#### **Enter**

Pressing key 3 clears the menu from the screen and the prompt at bottom left asks you to input the 'day' part of the date of the particular file. By now you'll have noticed the program uses a non-standard input routine, the length of the particular field being indicated by a row of star (\*) characters next to the prompt. As each character is typed it replaces one of the stars, the entry being complete either when the <RETURN> key is pressed or when the last character in that field is typed.

With date entries the data field is two characters wide, so a date comprising one digit (for example the '1' in 1/12/84) can be entered either as '1 < RETURN>' or as '01'. Once the day is entered you will be prompted for the month and then the year. To avoid unnecessary typing, the program allows you to 'borrow' date data from the last file you typed. Thus, if the date on the last record was '14/3/84' and you press the < RETURN> key in answer to the 'day' prompt in the next record, it will be assigned the date '14/3/84' as well. This can also be used with parts of dates, so if you input '28' for the day, then press < RETURN> when prompted to input the month, the date will be entered as '28/3/84'.

An error check routine is incorporated to detect illegal dates (including February 29 on non-leap years). Note, as each field is entered the partially complete file is displayed at the top of the display window.

After the date is entered you are asked to enter 'From/To'. This identifies from whom the money was received or to whom it was paid (maximum 16 characters). Pressing the <RETURN> key will enter the word 'Paid' in this field. The next item requested is the receipt or invoice number (maximum seven characters), which can be omitted by pressing the <RETURN> key, and the word 'B/card' inserted by pressing  $\uparrow$  B (that is, the <CONTROL> and 'B' keys simultaneously) or the word 'Cash' by pressing  $\uparrow$  C.

The next two steps deal with the category into which the item will be inserted in the palance sheet. The first asks you if it is a receipt or payment (press 'R' or 'P'), then all categories included in the ledger base will appear in the display window, with letters to identify them. Press the key corresponding to the particular category into which the payment falls. If you press a key that doesn't correspond to one of the displayed categories it will be ignored.

The final item to enter is the amount of money involved. Although this field is nine characters wide, two spaces are reserved for cents and one for the decimal place. (If you're dealing with amounts of a million dollars or more you can afford to get an accountant to do it for you!) There's no need to input trailing zeros in the cents part, so '100 dollars' can be entered as '100 <RETURN>' and '87 dollars 30 cents' as '87.3 <RETURN>'. Payments and receipts appear in separate right-justified columns in the display window: payments to the right and receipts to the left.

This completes your file entry and you will now be asked whether you wish to proceed to enter the next file (press < RETURN>), to return to the menu (key 'M') or to 're-do' (correct) the record you just typed (key 'R'). The program will not allow you to exceed the ledger capacity, so once the 320th file is typed it will return to the menu and announce 'Ledger Full'.

#### Save

This option allows you to save all or part of your ledger on tape. Pressing key 4 will take you into the 'SAVE' routine and you'll then be asked whether you want to save the base (key 1), the files (key 2) or the whole ledger (key 3). Pressing any other key aborts the routine, returning you to the menu.

In the majority of cases you will want to save the whole ledger on tape (key 3), but the base and file save facilities can be useful if you want to save only the files and incorporate a new base in the ledger with the 'Load' routine 'Merge' facility, or if you want to use the 'New Base' facility to

'Chain' a number of ledgers together (see below). Whatever option you choose, the program recognises the importance of your ledger files and saves two copies — just in case a fault in the tape causes a loading problem. The message 'Start Tape' will appear on the screen, followed by a short delay (to allow the tape leader to go past the record head), then the computer will announce it is saving 'Copy 1', 'Copy 2', then return to the menu.

The ledger 'header' (the important ledger information) is saved at 300 baud, then the base and files are saved at 1200 baud. This is not a job for bargain-basement tapes. I use TDK 120 us bias cassettes — anything less is just begging for an unloadable file that you'll have to type all over ag Database 'Save' routines tend to bring out the worst in Microbee casse... interfaces; readers whose Bees are 32 Kbyte IC models or earlier should consider making the tape I/O modifications suggested in the Microbee Hacker's Handbook (available through Your Computer), and those with the Telcom v1.0 or 1.1 communications ROM ought to either update to a later version or have the permanent RTC link disconnected. Otherwise, loading at 1200 baud becomes a game of chance!

#### Load

The load facility is designed for maximum flexibility in manipulating your ledgers. Not only can you load a full ledger (key 3) or a base or files alone (keys 1 or 2), but if there is already a ledger in memory and you choose to load only the files or base of your tape ledger, that section will replace the corresponding section of the ledger already in memory. This allows you to create a new base if the old one was deficient (such as if you decide an extra category ought to have been included in the payments section).

A 'Merge' facility (key 4) is also included, allowing you to add the files in your tape ledger to the end of the ledger (if any) already in memory. If this exceeds the ledger capacity (320 files), the computer announces "Merge-Part only", and you'll have to use the 'Display' function to check how many of the tape files were loaded before the ledger reached its capacity.

Once you've selected the load option you want to use, the program requests you to 'Start Tape', and this message remains on the screen until a valid tape header is encountered. At this stage the file title will be displayed with some additional information (such as whether the t consists of a base or files only or whether the files are unsorted — below). If the tape file is incompatible with your load command (such as if you commanded the computer to load a base and the tape dump is files only), the routine will abort leaving any existing ledger intact. Please note, limits on program length meant that no load indicator or error checking could be included, so if your ledger hasn't loaded after about two minutes then press <RESET> and try again.

#### Sort

This routine arranges the files in your ledger in chronological order according to the file dates. As LedgerMaster is written in BASIC the Sort routine is quite slow (about five minutes to sort 320 files), so you will be asked whether you are 'Sure?' Press key 'N' to abort or key 'Y' to continue.

Your files don't have to be sorted before they're saved on tape or printed, but your final printout will be much easier to follow if a sort is first carried out. A counter will appear on the screen as the routine is in progress (just to reassure you your computer hasn't died of a stroke!).

#### Delete

This function is useful if you want to get rid of an incorrectly typed file or a slab of files in a ledger that has been partly 'merged' with an earlier ledger. Enter the number of the first file to be deleted and the file will appear in the display window. Now press key 'N' if you've picked the wrong one or key 'Y' if it's correct. You'll then be asked for the number of the last file to be deleted. If only one file is to be deleted, press the <RETURN> key, otherwise enter the appropriate file number and check that the file that

appears on the display window is the last one in the section you want to erase. Both files that appeared on the screen will be deleted, along with all files in between. Invalid file numbers will be rejected.

#### Print

This is a powerful function with a range of options. You'll first be asked whether you want to search for any particular string, allowing you to search through the date, item, receipt number and/or money columns of the ledger for all occurrences of a particular set of characters. So if you entered 'B/card <RETURN>' in response to the 'Search' prompt, the only ligers printed would be those containing this expression.

If you're searching the money column for all entries of a particular amount, please remember trailing zeros in the cents column are omitted. A hash mark (#) followed by a file number entered in response to the search request will cause printing to commence at that file number. To print all files in the ledger simply press the <RETURN> key.

The program will now ask whether you want the printout directed to the screen or to a printer (keys 'S' or 'P'). If you select the printer option a submenu will appear, asking you to select the printer type (press an invalid key if you chose the printer option by mistake). Once you've made your selection you'll be asked to press a key to continue. Check your paper is properly aligned in the printer, press a key and away it will go.

The ledger printout consists of three sections (see Figure I). First are the categories 'carried forward' (if any), followed by the ledgers themselves and finally the balance sheet. Base categories with a nil balance are omitted. If the 'search' mode is used then only the ledgers are printed, as otherwise the balance sheet totals would be misleading.

When printing on the screen the ledger categories are shown simply by their code letter (to accommodate the 64-character-width screen), but these are printed in full if a printer is used. The printer format also includes a page heading showing the ledger name, column headings, page number and (if the ledger has been sorted) the dates it covers.

#### **New Base**

This option assists in 'chaining' ledgers by taking the base categories of an sting ledger, adding to each the total of all applicable files, then using the files themselves. This new base now contains 'carried forward' totals that are used in the new ledger, so its balance sheet reflects the total of the entries in both.

When the option is selected you'll first be asked if you're 'Sure?', as accidental selection of this routine would have disastrous results. You'll then be asked to enter the name of the new ledger, following which the necessary operations will be carried out and the program will return to the menu.

#### A Final Balance

No doubt some readers will have the utter horrors at typing in a program of this length. If you want to save yourself the trouble, forward a cheque or money order for \$15, payable to Dreamcards, 8 Highland Court, North Eltham 3095 (mail order only), and I'll send you a tape.

Anyone wanting a more sophisticated tape-based accounting system might like to consider my Multibank program (\$34.95 plus \$1.50 postage and packing, for cassette and manual). This generates cassette files containing up to eight ledgers, and has heaps of frills usually only found on expensive disk-based systems (such as automatic allowance for periodical debits).

#### **Coming Soon**

Next month's issue of *Your Computer* will tell you how LedgerMaster works and how to convert it to run with other BASICs.

# LedgerMaster

```
00001 POKE 140,1 REM Disable <BREAK> Key
00001 POKE 140.1 REM Disable <BREAK> Key
00002 GOSUB 333 REM Initialise Storage arrays
00003 IN#0: OUT#0 REM VDU On
00004 N=8: IF BO$(0)="" THEN LET N=2 ELSE IF C=1 THEN LET N=4
00005 KO$=KEY$: K4$="": I=0: Q=0: IF C<3 AND N=6 THEN LET N=5
00006 GOSUB 298: INVERSE: CURS 28.4: PRINT "MENUL": NORMAL
00007 PRINT \SPC(14) "Create^\.__1";
00008 PRINT SPC(14) "Load^\.__2": IF N=2 THEN 16
00009 PRINT SPC(14) "Enter^\.__3";
00010 PRINT SPC(14) "Bate^\.__3";
00010 PRINT SPC(14) "Bate^\.__3";
00011 PRINT SPC(14) "Delete^\.__3";
10011 PRINT SPC(14) "Sot^\._6"
00012 PRINT SPC(14) "Sot^\._6"
00013 PRINT SPC(14) "Print^\.__7";
00014 PRINT SPC(14) "Print^\.__7";
00015 IF C=V+1 THEN PRINT \SPC(21) "****LEDGER FULL^***"
 00014 PKINT SPC(4) "New Base"..."8"

00015 IF C-V-1 THEN PRINT \ SPC(21) "****LEDGER*FULL****"

00016 K2$="Select"Menu^Option": GOSUB 226

00017 X=INT(VAL(KO$)): IF X=0 OR X>N THEN 16

00018 PLAY 22.1: ON X GOTO 65.33,83.120.188.22,150,216
  00019 REM
                                         ----- Sort Routine -----
  00020 REM -
  00021 REM
  00022 GOSUB 232: IF K1$="N" OR S=1 THEN 3
 00022 GOSUB 222: IF AI.S= N UN S=1 IHEN 3
00023 X=0: GOSUB 317 REM Set-up Date array
00024 FOR X=1 TO C-1: GOSUB 292: A1(X)=F4: NEXT X
00025 FOR X=1 TO C-2: GOSUB 317: FOR M=X+1 TO C-1 REM SORT
00026 IF A1(M)=>A1(X) THEN 29
  00027 K0$=A0$(M): A0$(M)=A0$(X): A0$(X)=K0$
00028 F1=A1(M): A1(M)=A1(X): A1(X)=F1
  00029 NEXT M: NEXT X: S=1: GOTO 79
  00030 REM
  00031 REM ----- "Load" Routine -----
  00032 REM
  00033 N=3: IF C>1 AND C<V+1 THEN LET N=4
 00033 N=3: IF C>1 AND C<V+1 THEN LET N=4
00034 GOSUB 324: IF 0-0 THEN 3
00035 OUT#0 OFF: IN#2 REM VDU Off, 300 baud Cass. in
00036 INPUT K1$, D. H. X. Y. Z. E: K1$=K1$(:7)
00037 IN#0: OUT#0 REM VDU On again
00038 IF 0-3 AND H<3 THEN LET 0-0
  00039 IF 0=1 AND H=2 THEN LET 0=0
00040 IF 0>1 AND H=1 THEN LET 0=0
00040 IF 0-1 AND H-1 THEN LET 0-0
00041 K0$="": J=1: L=D-1: IF 0-4 THEN 44
00042 J=C: L=L+C: C=L: S=0: K0$=""-"Merge"
00043 IF C>V+1 THEN LET C=V+1: K0$=K0$+""(Part^only)"
00044 IF 0-2 OR 0-3 THEN LET C=D: S-Z
00045 IF 0-1 OR 0-3 THEN LET B0$(0)-K1$: P=X: R=Y: T=E
00046 K1$=K1$+K0$: CURS 0,12: PRINT [A64^32]
00047 CURS (64-LEN(K1$))/2,12: PRINT K1$: CURS 26,13
00048 IF H=1 OR 0-1 OR C-1 THEN PRINT ""(Base)": GOTO 51
00049 IF (H=2 OR 0-2) AND C>1 THEN PRINT ""(Ipie)": GOTO 51
00050 IF S=0 AND C>1 AND O>1 THEN PRINT "(Unsorted)"
00051 IF 0-0 THEN 79 ELSE LET N=X: IF Y>X THEN LET N=Y
00052 CURS 0: OUT$0 OFF: IN$3 REM VDU Off, 1200 baud Cass. in
00053 FOR X=1 TO N REM Load Base
00054 IF (0-2 OR 0-4) AND H-3 THEN INPUT K0$, K0$, F1, F1
00055 IF 0-1 OR 0-3 THEN INPUT B0$(X), B1$(X), B2(X), B3(X)
00057 INPUT A0$(X), A0$(X+1), A0$(X+2): NEXT X
 00057 INPUT AO$(X), AO$(X+1), AO$(X-2): NEXT X
00058 IF C<V+1 AND O>1 THEN GOSUB 336
00059 Y=P+1: IF P<20 THEN GOSUB 337
00060 Y=R+1: IF R<20 THEN GOSUB 338
  00061 GOTO 79
  00062 REM
  00063 REM ----- "Create" Routine -----
  00064 REM
  00065 GOSUB 331: IF K1$="N" THEN 3
 00066 G-24 K25-"[16-Title": GOSUB 298: GOSUB 250: B0$(0)=K1$: I=1
00067 GOSUB 304: P=P-1: GOSUB 249: B0$(P)=K1$: K35=K1$
00068 IF K3$-"" AND P-1 THEN LET B0$(1)="Payment": GOTO 70
00069 IF K3$-"" THEN LET P=P-1: GOTO 71
  00070 GOSUB 237: B2(P)=F3: IF P<20 AND K3$<>"" THEN 67
  00071 PLAY 22.1
 00071 PLAY 22.1
00072 GOSUB 307: R=R+1: GOSUB 249: B1$(R)=K1$: K3$=K1$
00073 IF K3$="" AND R=1 THEN LET B1$(1)="Receipt": GOTO 75
00074 IF K3$="" THEN LET R=R-1: GOTO 76
00075 GOSUB 237: B3(R)=F3: IF R<20 AND K3$<>"" THEN 72
00076 PLAY 22.1: T=0: FOR X=1 TO 20: IF B2(X)>0 THEN LET T=T+1
00077 IF B3(X)>0 THEN LET T=T+1
00078 NEXT X: C=1+ K2$=""." COTO 2
  00078 NEXT X: C=1: K3$="":
                                                                                  GOTO 3
  00079 CURS 0: PLAY 22,1; 0,5: GOTO 3
  00080 REM
  00081 REM ----- "Enter" Routine -----
  00082 REM
  00083 IF C>V THEN 3
00084 S=0: AO$(C)="": M=C: GOSUB 268: I=C-1: G=2
00085 IF C>1 THEN LET X=C-1: GOSUB 292
```

# LedgerMaster

```
00171 D=2: PRINT "LEDGER:^"; BO$(0); SPC(26-LEN(BO$(0)));
00172 K2$="'to^": IF S=0 THEN 175 REM Print dates if SORTED
00173 K0$=A0$(1): GOSUB 286: K2$=K1$-K2$
00174 K0$=A0$(C-1): GOSUB 286: K2$=K1$-K2$
00175 PRINT SPC(34-LEN(K2$)): "Page^": F4: F4=F4+1
00176 PRINT "No.^^DATE^^TRANSACTION"; SPC(17);
00177 PRINT "CLASS"; SPC(16); "AMOUNT^($)": PRINT [A79^45]
00178 IF 0=0 OR K4$<>"" THEN 188 REM Print "Carried Forward"
00179 U=U+1: IF U>P THEN 182
00180 F1=B2(U): IF F1=0 THEN 179
00181 K0$=R0$K(U): X=LENK(K0$): Z=0: GOTO 184
 00086 K2$="Day": GOSUB 250: IF K1$="" AND F1>0 THEN 92
00087 F1=VAL(K1$): IF F1=0 OR F1>31 THEN 86
00088 K2$="Month": GOSUB 250: IF K1$="" AND F2>0 THEN 92
00089 F2>VAL(K1$): IF F2=0 OR F2>12 THEN 88
00090 K2$="Year": GOSUB 250: IF K1$="" AND F3>0 THEN 92
00091 F3=VAL(K1$): IF F3=0 THEN 90
0092 RESTORE: FOR X=1 TO INT(F2): READ Z: NEXT X
00093 DATA 31, 28, 31, 30, 31, 30, 31, 30, 31, 30, 31
  00093 DAIA 31,28,31,30,31,30,31,31,30,31,30,31
00094 IF FRACT(FS/4)=0 AND F2=2 THEN LET Z=29
00095 IF INT(F1)<-Z THEN 97
00096 CURS 1,16: PRINT "****DATE^ERROR^****;: PLAY 10,12: GOTO 86
00097 KO$=$TR$(INT(F1)): K1$=KO$(;2)+"/"
                                                                                                                                                                                                                                                                            00181 K0$=B0$(U): X=LEN(K0$): Z=0: GOTO 184
00182 W=W+1: F1=B3(W): IF F1=0 THEN 182
                                                                                                                                                                                                                                                                          00182 W-W+1: F1-B3(W): IF F1-O THEN 182
00183 KO$-B1$(W): X=LEN(KO$): Z-11
00184 IF O-T THEN PRINT "CARRIED FORWARD:^^"; ELSE PRINT SPC(20);
00185 IF Q-O THEN PRINT SPC(22): KO$; SPC(27-X-Z);
00186 IF Q-O THEN PRINT KO$: SPC(33-X-Z);
00187 O-O-1: PRINT [F10.2*F1]: GOTO 205 REM Print amount carried
00188 IF K4$="" OR K4$="#" OR O<>1 THEN 190
00189 PRINT \ SPC(10); "SEARCH;"; K4$ \: O=0: G=3: GOTO 205
00190 M-H: IF K4$="" THEN LET M-H-T
00191 IF M<1 OR M>-C THEN 194 ELSE GOSUB 269 REM Print FILE
00192 IF M<0 THEN LET BS(N)-F1 ELSE LET B4(N)-F4 IN)-F1
   00098 KO$=STR$(INT(F2)): K1$=K1$+K0$(;2)+"/"
00099 KO$=STR$(INT(F3)): IF F3<10 THEN LET K1$=K1$+"0"
  00099 K0$=STR$(INT(F3)): IF F3<10 THEN LET K1$=K1$*"0"
00100 K1$=K1$+K0$(:2): GOSUB 267: K2$="From/To": G=16: I=1
00101 GOSUB 250: IF K1$="" THEN LET K1$="Paid"
00102 GOSUB 267: K2$="Cheque/Receipt"No.": G=7: GOSUB 250
00103 IF X=2 THEN LET K1$="B/card" ELSE IF X=3 THEN LET K1$="Cash"
00104 GOSUB 267: K2$="Receipt"(R)^or"Payment"(P)"
00105 GOSUB 226: IF K1$<>"R" AND K1$<>"P" THEN 105
00106 GOSUB 285: GOSUB 298
00107 IF K1$<>"P" THEN 109
00108 M=P: IF M=1 THEN LET K1$="A": GOTO 111 ELSE GOSUB 304: GOTO 110
00109 M=R: IF M=1 THEN LET K1$="A": GOTO 111 ELSE GOSUB 307
00110 K2$="Category": GOSUB 226: X=64: IF X<1 OR X>M THEN 110
                                                                                                                                                                                                                                                                            00192 IF M=0 THEN LET B5(N)=B5(N)+F1 ELSE LET B4(N)=B4(N)+F1
                                                                                                                                                                                                                                                                            00193 GOTO 205 REM Add balance in file to BASE category
00194 IF K4$<>"" THEN 205
00195 M=M-C: IF M=0 THEN PRINT: GOTO 205 REM FILE End
00196 IF M-P THEN 199 ELSE IF B4(M)=0 THEN 209 REM Payments
00197 PRINT SPC(20): BO$(M): SPC(22-LEN(BO$(M))); [F10.2°B4(M)]
   00110 K2$="Category": GOSUB 226: X=X-64: IF X<1 OR X>M THEN 110
  00111 GOSUB 267

00112 G-9: I =0: K2$="Amount": GOSUB 250: GOSUB 242: IF X=1 THEN 112

00113 GOSUB 267: K2$="Next-<CR>^"Menu=M^"Re-do=R^":I=1

00114 GOSUB 226: IF K1$="R" THEN 84

00115 IF K1$<>"W" AND X<>128 THEN 114

00116 C-C+1: IF X=128 THEN 83 ELSE 3
                                                                                                                                                                                                                                                                          00197 PRINT SPC(20): BO$(M): SPC(22-LEN(BO$(M))); [F10.2"B4(M)]
00198 F3=F3-B4(M): GOTO 205
00199 M=M-P: IF MSR THEN 202 ELSE IF B5(M)=0 THEN 209 REM Receipts
00200 PRINT SPC(20): B1$(M): SPC(22-LEN(B1$(M))); [F10.2"B5(M)]
00201 F3=F3+B5(M): GOTO 205
0202 X=67: IF Q=0 THEN LET X=51 REM Print Total at end
00203 IF H=E OR H=E-2 THEN PRINT SPC(X+2); [A10"45]: G=1
00204 IF H=E-1 THEN PRINT SPC(X): [F11.2"R3]: G=1
00205 D=D+G: IF D<L THEN 209 REM Lines per page counter
00206 REM Delete next line if not using fan fold paper
00207 IF Q>0 THEN PRINT \\\\\\\\\\\\\\\\\ REM Perf. gap = 7 lines
00208 D=O: IF Q>0 THEN 209 ELSE GOSUB 225: IF KO$="A" THEN 212
00209 H=H+1: IF H<E-T THEN 166 REM Detect end of printing
00210 IF Q>0 THEN 0UT≠Q OFF: GOTO 212
00211 IF KO$<"A" AND KO$<>"a" THEN GOSUB 225
   00118 REM ----- "Save" Routine -----
   00119 REM
00120 N=3: IF C=1 THEN LET N=1
   00121 GOSUB 324: IF H=0 THEN 3
00122 N=P: IF R>P THEN LET N=R
00123 FOR Q=1 TO 2: PLAY 0.60: KO$=BO$(0)
00124 CURS 0.12: PRINT [A28^32]: "Copy": Q: |A28^32|: CURS 0
00125 OUT#2 REM 300 Baud Cassette out (VDU remains on)
00126 PRINT "******* KO$: ".": C: ".": H: ".": P: ".": R: ".": R: ".": T
00127 PLAY 0.1: OUT#3 REM Delay / 1200 Baud Cassette out
00128 IF H=2 THEN 130 ELSE FOR X=1 TO N REM Save BASE
00129 PRINT BO$(X); ".": BI$(X): ".": B2(X): ".": B3(X): NEXT X
00130 PLAY 0.1: IF C<2 OR H=1 THEN 133
00131 FOR X=1 TO C STEP 3 REM Save FILES
00132 PRINT AO$(X): ".": AO$(X+1): ".": AO$(X+2): NEXT X
00133 OUT#0 REM De-select Cassette output
00134 NEXT Q: GOTO 79
00135 REM
   00123 FOR Q=1 TO 2: PLAY 0,60: KO$=BO$(0)
                                                                                                                                                                                                                                                                            00212 Q=0: GOTO 3
00213 REM
                                                                                                                                                                                                                                                                            00214 REM
                                                                                                                                                                                                                                                                                                                ----- "New Base" Routine
                                                                                                                                                                                                                                                                            00215 REM
                                                                                                                                                                                                                                                                            00216 GOSUB 232: IF K1$="N" THEN 3

00217 I=1: G=24: K2$="New^Title": GOSUB 250: IF K1$<>"" THEN LET BO$(0)=K1$

00218 FOR Y=1 TO C-1: K0$=A0$(Y): X=SEARCH(K0$,"|",3): GOSUB 287

00219 0-ASC(K0$): GOSUB 286: Z=ASC(K0$)-64: GOSUB 286: F1=VAL(K0$)

00220 IF 0-80 THEN LET B2(Z)=B2(Z)+F1 ELSE LET B3(Z)=B3(Z)+F1
   00135 REM
                                                                                                                                                                                                                                                                            00221 X=Y: GOSUB 317: NEXT Y: C=1: GOSUB 336: GOTO 76
                                                     ----- "Delete" Routine
   00136 REM --
   00137 REM
                                                                                                                                                                                                                                                                            00222 REM
   00138 G=3: K3$="^file^to^be^deleted": K2$="First"-K3$
                                                                                                                                                                                                                                                                            00223 REM ----- "Key to Continue" Sub. -----
 00138 G-3: K3$="file^to^be^deleted": K2$="First"-K3$
00139 GOSUB 250: O=INT(VAL(K1$)): IF O<1 OR O=>C THEN 138
00140 M=0: GOSUB 268: GOSUB 232: IF K1$="N" THEN 3
00141 G-3: K2$="Last"+K3$+""-for*CCR>": I=1: GOSUB 250
00142 L=INT(VAL(K1$): IF K1$="" THEN LET L=0
00143 I=0: IF L<0 OR L=>C THEN 141 ELSE IF L=0 THEN 145
00144 M=L: GOSUB 268: GOSUB 232: IF K1$="N" THEN 141
00145 K3$="": Z=L=O+1: X=0: GOSUB 316: GOTO 79
                                                                                                                                                                                                                                                                            00224 REM
                                                                                                                                                                                                                                                                            00225 K2$="KEY^to^Continue
                                                                                                                                                                                                                                                                            00226 G=1: GOSUB 250: X=ASC(K1$)
                                                                                                                                                                                                                                                                            00227 IF X>96 AND X<123 THEN LET X=X-32: K1$=CHR$(X)
                                                                                                                                                                                                                                                                            00229 REM
00230 REM
                                                                                                                                                                                                                                                                                                                 ----- "Sure (Y/N)" Sub. ----
                                                                                                                                                                                                                                                                            00231 REM
                                                                                                                                                                                                                                                                            00232 K2$="Sure^(Y/N)": GOSUB 226: IF K1$<>"N" AND K1$<>"Y" THEN 232
   00147 REM
                                                                       ····-- "Print" Routine
                                                                                                                                                                                                                                                                            00233 RETURN
                                                                                                                                                                                                                                                                            00234 REM
   00149 REM
00149 KEM
00150 G=20: I=1: H=1: K2$="String"search": GOSUB 250: K4$-K1$
00151 IF ASC(K4$)=35 THEN LET H=INT(VAL(K4$(:2))): K4$="#"
00152 IF H=0 OR H>C-2 THEN 150
00153 K2$="Screen"(S)"or"Printer"(P)": GOSUB 226
00154 L=12: IF K1$="S" THEN 163 ELSE IF K1$<"P" THEN 3
00155 GOSUB 298: K2$="PRINTERTType:": PRINT \\ SPC(15): K2$
00156 PRINT \SPC(20) "Paralle1"\\ "FKY"1"
00157 PRINT SPC(20) "Serial 1300bd1"\"FKY"1"
00158 PRINT SPC(20) "Serial 1300bd1"\"FKY"2"
00158 PRINT SPC(20) "Serial 1200bd1\"FKY"3"
00159 PRINT SPC(20) "Serial 1200bd1\"FKY"3"
00159 PRINT SPC(20) "Serial 1200bd1\"FKY"3"
00160 GOSUB 226: X=4-48: IF X-0 OR X>4 THEN 3
00161 L=57: Q=X: IF X>1 THEN LET Q=X-2
00162 GOSUB 226: X=4-8: IF X>1 THEN LET Q=X-2
00163 FOR X=1 TO 20: B4(X)=B2(X): B5(X)=B3(X): NEXT X
00164 D=0: F3-0: F4-1: O=T: U=0: W=0: E=C-P+R+T-4
00165 IF K4$<>"THEN LET E=C: O=1
00166 G=1: K0$=K8Y$: IF K0$="" THEN 169 ELSE PLAY 22.2
00167 IF K0$="A" OR K0$="" THEN 169 ELSE PLAY 22.2
00168 K1$=KEY$: IF K1$="" THEN 168 REM Hold if any other key
00169 IF D>0 THEN 178 ELSE IF Q>0 THEN 171 REM Page Heading
00170 FOR X=3 TO 14: CURS 1,X: PRINT [A64*32]: NEXT X: CURS 1,3: GOTO 178
   00150 G=20: I=1: H=1: K2$="String^search": GOSUB 250: K4$*K1$
                                                                                                                                                                                                                                                                            00235 REM ----- "Previous Balance" Input Sub. -----
                                                                                                                                                                                                                                                                            00236 REM
                                                                                                                                                                                                                                                                            00237 G=9: K2$="Balance"Forward": GOSUB 250: GOSUB 242
00238 IF X=1 THEN 237 ELSE RETURN REM Loop if invalid amount
                                                                                                                                                                                                                                                                            00239 REM
                                                                                                                                                                                                                                                                            00240 REM ----- "Money Input" Sub. ----
                                                                                                                                                                                                                                                                            00241 REM
                                                                                                                                                                                                                                                                            00242 X=1: F1=VAL(K1$): IF F1<=0 AND I=0 OR F1=>1000000 THEN 245
                                                                                                                                                                                                                                                                            00243 X=0: F2=FRACT(F1): F3=F1-F2+FLT(INT(F2*100))/100
00244 K1$=STR$(F3): K1$=K1$(:2) REM Round off to 2 places
                                                                                                                                                                                                                                                                            00245 RETURN
                                                                                                                                                                                                                                                                            00247 REM ----- Main "Input" Sub. -----
                                                                                                                                                                                                                                                                            00249 G=12: K2$="Category
                                                                                                                                                                                                                                                                           00259 GSUB 299: Y=LEN(K2$): CURS 1,15: PRINT [A64^45] [A63^32];
00251 X=V+1-C: IF X>V THEN LET X=V REM Limit free files counter
00252 CURS 1,16: PRINT K2$;: CURS 52,16: PRINT INT(FRE($)); "":"; X:
00253 CURS Y+3,16: Z=0: K1$="": FOR X=1 TO G: PRINT "*":: NEXT X
00254 CURS Y+1,16: PRINT "?"; REM Print prompt line
00255 K0$=KEY$: X=ASC(K0$): IF X=124 OR X=128 THEN 255
```

```
00256 IF X=8 OR X=127 THEN 261 ELSE IF X>13 THEN 259
00256 IF X=8 UN X=127 THEN 261 ELSE IF X>13 THEN 259
00257 IF Z>0 THEN CURS O: RETURN REM End if <CR>
00258 IF I>0 THEN LET K1$="": RETURN ELSE 255
00259 Z=Z+1: CURS Y+Z+2.16: PRINT K0$: REM Print Input so far
00260 K1$=K1$+K0$: IF Z=G THEN RETURN ELSE 255 REM End of string?
00261 K0$=K1$(1, LEN(K1$)-1): K1$=K0$ REM <DEL>ete Key
00262 IF Z>0 THEN CURS Y+Z+2.16: PRINT "*";: CURS Y+2.16: Z=Z-1
00263 GOTO 255
00264 REM
00265 REM ----- "File entry" Print Sub. ----- 00266 REM
00267 M=C: GOSUB 285 REM "M" is record number
00268 GOSUB 298: PRINT "FILE:"
00269 IF K4$="" OR K4$="*" THEN 271 REM Implement "Search"
00269 IF K4$="" OR K4$="*" THEN 271 REM Implement "Search"
00270 K0$=A0$(M): IF SEARCH(K0$, K4$)=0 THEN LET G=0: RETURN
00271 K0$=STR$(M)+".": K0$=K0$(:2): X=LEN(K0$)
00272 PRINT K0$; SPC(4-X): K0$=A0$(M): GOSUB 286 REM Date
00273 PRINT SPC(9-LEN(K1$)): K1$: """: IF Q>0 THEN PRINT """:
00274 GOSUB 286: PRINT K1$; SPC(18-X): REM Item
00275 GOSUB 286: PRINT K1$; SPC(18-X): REM Cheque/Receipt No.
00276 GOSUB 286: M=0: IF K1$="P" THEN LET M=11 REM Rec/Pay?
00277 IF Q=0 THEN PRINT K1$: "": REM Don't print in full on VDU
00278 GOSUB 286: N=ASC(K1$)-64: K2$=B0$(N): IF M=0 THEN LET K2$=B1$(N)
00279 IF Q>0 THEN LET X=LEN(K2$): PRINT K2$; SPC(16-X): ELSE PRINT K1$:
 00280 GOSUB 286: F1=VAL(K1$): IF F1>0 THEN PRINT SPC(M); [F10.2^F1]
00281 RETURN
 00282 REM
 00283 REM ----- Extract file section Sub. -----
 00284 REM
00285 ACM$(C)=A0$(C)+K1$+"|": RETURN REM "|" delimits section
00286 ACM$="": X-SEARCH(K0$,"|"): IF X=0 THEN RETURN
00287 K1$=K0$(:1,X-1): K0$=K0$(:X+1): IF Q>0 THEN LET X=X-1
 00288 RETURN
 00290 REM ----- Extract Date Variables Sub. -----
 00291 REM
00292 K1$=A0$(X): F1=VAL(K1$): Y=SEARCH(K1$,"/")+1: K1$=K1$(;Y)
00293 F2=VAL(K1$): Y=SEARCH(K1$,"/")+1: K1$=K1$(;Y)
00294 F3=VAL(K1$): F4=100*F3 + F2 + F1/100: RETURN
 00296 REM ------ Print Screen heading Sub. -----
 00298 CLS
 00299 CURS 17.1: INVERSE: PRINT "^*"Dreamcards LedgerMaster *" 00300 NORMAL: PRINT [A64 45]:: RETURN
 00302 REM ----- Print BASE Sub. -----
 00303 REM
 00304 GOSUB 298: PRINT "PAYMENT^CATEGORIES:^^("; BO$(0); ")": K2$="Category"
 00305 IF P=0 THEN RETURN REM Payments
00306 FOR X=1 TO P: KO$=BO$(X): F1=B2(X): GOSUB 310: NEXT X: RETURN
00306 FOR X=1 TO P: KO$=BO$(X): FI=B2(X): GO$UB 310: NEXT X: RETURN 00307 GO$UB 298: PRINT "RECEIPTS^CATEGORIES:^^("; BO$(0); ")" 00308 IF R=0 THEN RETURN REM Receipts 00309 FOR X=1 TO R: KO$=B1$(X): FI=B3(X): GO$UB 310: NEXT X: RETURN 00310 J=X+3: K-3: IF X>10 THEN LET J=J-10: K-35 00311 CURS K,J: PRINT CHR$(X+64); "."; KO$: $PC(13-LEN(KO$)): 00312 IF FI>0 THEN PRINT [F10.2°F1] REM Print any balance
 00313 RETURN
 00314 REM 00315 REM ----- "Counter" Print Sub. -----
 00316 REM
 00317 IF X>1 THEN 319
 00318 CURS 1.12: PRINT [A26^32] "*^WAIT^*" [A92^32] 00319 IF X>0 THEN CURS 28,13: PRINT [I4^X];
 00320 CURS 0: RETURN
 00321 REM
 00322 REM ----- Tape I/O Prompt Sub. -----
 00323 REM
00324 K25="Base^(1)": IF N>2 THEN LET K2$=K2$+", Files^(2), All^(3)"
00325 IF N>3 THEN LET K2$=K2$+", Merge^(4)"
00326 GOSUB 226: H=INT(VAL(K1$)): IF H>N THEN LET H=0
00327 O-H: CURS 26.12: PRINT "Start tape": CURS 0: RETURN
 00328 REM
 00329 REM ----- Initialise Arrays Sub. -----
 00330 REM
 00331 IF C=0 THEN 333 REM Print prompt if existing file
 00332 GOSUB 232: IF K1$="N" THEN RETURN
00333 CLEAR: V=320: STRS(V*51+1400) REM V = Max. No. of files
00334 DIM AO(V+2), AI(V+2), BO(20), B1(20), B2(20)

00335 DIM B3(20), B4(20), B5(20): GOSUB 337: GOSUB 338

00336 FOR X=C TO V+2: AO$(X)="": NEXT X: K1$="": RETURN

00337 FOR X=Y TO 20: B0$(X)="": B2(X)=0: NEXT X: RETURN

00338 FOR X=Y TO 20: B1$(X)="": B3(X)=0: NEXT X: RETURN
```

# "CHARLIE'S CASHROOK

#### Any "Charlie" Can Use It!

"Cashbook" is two programs in one . . . both working together. The first manages all your bank accounts and the second manages

your expense ledgers.
"Cashbook" will save hours of your accountant's valuable time and provide 'up-to-date' financial reports . . . whenever you require them!

# But, What Does It Do? • Ultra fast

- Multi-Company or Department
   Multiple Bank Bank Accounts
   Detailed Bank Reports
   Detailed Expense Reports

- Full Bank reconciliation
- Comprehensive documentation
- Runs on IBM and compatibles

**OFFER:** 

SPECIAL THROUGH NOW! ... AND WE THROUGH NOW! ... AND WE WILL HAVE "CASHBOOK" ON YOUR DESK IN THE MORNING

#### **Charlie's Computer Products**





★ Dealer Enquiries Welcomed ★



P.O. Box 513, South Perth, W.A. 6151 PHONE: (09) 367-7532



#### Rockwell **QUALITY SEMICONDUCTOR PRODUCTS** 68000 - 650**0**

R68000C6	CPU Ceramic DIL	\$55.79
R68000Q6	CPU Plastic Compact QUIP	\$36.76
	(8, 10 MHz avail. 12 MHz avail. soon)	
R6502P	ČPU, 40 pin	\$8.29
R6503P	R6504P R6507P CPU, 28 pin	\$7.24
R6511Q	Single chip microcomputer, 64 pin CPU, RAM ACIA I/O,	***
DOFOOD	Timers, 64k Address	\$20.93
R6520P	PIA 40 pin	\$5.13
R6522P	VIA 40 pin, 20 I/O, 2 × Timer/Ctr S. Reg	\$7.37
R6532P	COMBO 40 pin, 128 byte RAM, 20 I/O, timer	\$9.21
R6541Q	Slave Processor, 64 × 8 RAM, 23 I/O, host slave I/F, timers, 4K	
	external address	\$15.01
R6545-1P	Video controller	\$12.51
R6551P	ACIA integral Baud rate gener	\$8.82
R6592P	Single chip printer controller for Epson Low Cost Printer	\$15.14
CMOS	•	
R65C02 P1	CPU, 40 pin plastic, enhanced instruction set	\$10.00
R65C21 P1	PIA 40 pin	\$7.90
R65C24 P1	PIAT, I/O × Timer 40 pin 6821/6521 compatible	\$9.48
R65C51 P1	ACIA 28 pin	\$10.00

NOTE: Plus \$5.00 P & P. All prices Tax exempt

Note: Rockwell CMOS has superior latch up resistance compared to compatible designs from other makers using first generation masks. PARTS ALSO AVAILABLE 2MHz

CONTACT THE OFFICE FOR PRICES AND AVAILABILITY



All prices are subject to change without notification. For immediate despatch of your order, phone and quote your Bankcard number. SUSTANTIAL OEM QUANTITY DISCOUNTS AVAILABLE.

73 Eric Street, Goodna, Qld., Australia P.O. Box 6502, Goodna, Qld. 4300 Telephone (07) 288 2455. Telex AA 43778 ENECON

# CB80/86 Standard I/O Library

The lengthy CBASIC Compiler (CB80 and CB86) listings here may seem a bit much for a single program — and they are. We're including our in-house function library, and its support files, both to make your own programming easier and to ensure you have the library available in future when we publish other programs which use it.

SEVERAL OF THE FUNCTIONS listed in the following pages are based on a set of PL/I routines written by John Skaller for our subscriptions system. They are in many ways an effective replacement for Digital Research's Display Manager — John had to write them because our DM86 hadn't arrived by the time we had to have our system running.

The dBase file fixer FIXDBF.BAS uses several of the functions in the library, but certainly not all of them — you can be selective about those you type in if you wish. However, we recommend you take the complete library, and compile and LIB it for easy use in the future.

The files which make up the function library are these

STDIO.BAS: The function definitions themselves (we'll explain them one-by-one later). These can be used as is by %INCLUDEing them in your source file, or defined as PUBLIC for library creation and an appropriate set of EXTERNAL declarations created.

STDIO.DCL: These are declarations of the global variables used by the functions (and by the programs themselves). All are declared COMMON, to allow for the separate library approach.

STDIO.ASS: This file assigns values to the variables declared in the DCL file, and is normally %INCLUDEd at the start of your source file.

STDIO.TRM: Most of the functions use direct screen addressing and other terminal-specific control codes. The default values in the library are for terminals in the Televideo/ADM style, which may not suit you. Alternative codes can be put in this file, and a call to the GET.PARMS function will install these values in your program.

Now, let's look at the functions themselves:

**Unbug(v\$)** is useful for debugging a program — it is passed a message which is printed to the status line of the TVI950 terminal.

Conout(v\$), coninp\$, and conprt(msg\$) are simple 'print a character', 'get a character' and 'print a string' routines used by the I/O functions. We have versions which use BDOS calls, but suggest you use these simplified ones for portability.

Terminit, highlight, lowlight, reverse, unrevrs, clrscr, clreol and bell are self-explanatory.

Gotoxy(x%,y%) allows direct screen addressing. It is fed x and y co-

ordinates, which should be 1 to 24 and 1 to 80 respectively. If it is given a co-ordinate of 0, it assumes the last-addressed position. (There is no provision for terminals which send column before row.)

Displ(x%,y%,v\$) displays the string v\$ at the co-ordinates specified, while displyl(x%,y%,v\$) does the same and clears to end-of-line.

Fmtdate\$(da\$) turns a date string in the form 850501 around to

Fmtnum\$(de,le%) takes a number and turns it into a string of a specified length to allow easier screen editing.

Fmtdol\$(de,le%) does the same, but in decimal format (that is, it shows 1000 as 10.00).

Valid\$(q\$,type%) is used by the input functions to validate the operator's input, which can be forced to alpha, numeric, upper case or only alpha

Vinput\$(x%,y%,n%,type%) is a string input routine. At the x-y position specified it puts n% dots (the length of the string to be input) and accepts only type% input. Backspacing will delete the previous character and replace it with a dot. Valid type% values are alpha, numeric, upper and onlyalpha (assigned in STDIO.ASS). Linput\$(x%,y%,n%) is a simpler way to use vinput — it assumes a type of alpha.

Vupd\$(x%,y%,deflt\$,type%) is a string update routine which works in much the same way as Vinput, but with an existing default. Lupd\$ (x%,y%,deflt\$), like Linput, assumes a type of alpha.

Syserr(x%,y%,v\$) prints an error message v\$ at the specified co-ordinates and waits for a key to be hit before continuing.

Yesno%(x%,y%,v\$,d%) puts the question v\$ at the specified position, follows it with 'Y/N', and positions the cursor on the default answer (d%). The default can be yes, no, or none — if it is none the cursor is positioned on the slash and an answer of Y or N is forced.

Verify%(tmpstr\$,patrn\$) compares a string to a given pattern to confirm that only legal characters (that is, those in the pattern) are include:

Vdate%(vddmmyy\$) verifies a date passed as ddmmyy, Revdate\$(mmyy\$) reverses a date from ddmmyy to yymmdd or vice versa.

Ninput(x%,y%,fln%,minval,maxval,defval) is like Vinput, but for numbers, and allows you to specify length (fln%), minimum, maximum and default values.

Nupd(x%,y%,fln%,minval,maxval,defval,deflt) provides numeric update, like Lupd, but allows for an alternative default. Note that in Ninput and Nupd all numbers are converted to strings for the sake of convenient screen handling, and converted back again later.

Doscan%(patrn\$,base\$) checks how often the patrn\$ occurs in the string base\$. It is a support routine for Isvalid\$. Both functions are adapted from a Digital Research sample program.

Isvalid\$(fname\$) checks the validity of a filename, returning a null string if it is incorrect.

If it is incorrect.

Chrconv\$(v\$) is a support routine for Get.parms. It converts decimal

ASCII values to the appropriate characters for screen control sequences. **Get.parms** allows you to change screen control strings to suit different terminals. It looks on the disk for a file called STDIO.TRM and, if it finds it, replaces the values in memory with the new values.

Trim\$(v\$) trims trailing blanks from the string v\$

**Delay(length)** provides a delay of approximately *length* seconds. This depends on dlyfactor, set in STDIO.ASS, which will vary from machine to machine. The default of 250 works on a 6 MHz 8085.

**Err.handler** is an error handler which simply prints the error code and exits. It assumes the error is fatal, and is provided mainly for use during the development phase of a program.

```
next qi%
ccol=ccol+len(v$)
def unbug(v$)
   rem print to status line of kokusai print escape+"g"+escape+"f"+v$+chr$(13);
                                                      _____CLRFOL
    ko%=inkey
                                                     def clreol
                                                         call conprt(eraeol)
                                                     fend
def conout(v$)
print v$; fend
                                                     def displyl(x%,y%,v$)
    call displ(x%,y%,v$)
    call clreol
aef coninp$
    v$=chr$(inkey)
                                                     coninp$=v$
                                                     def clrscr
fend
                                                         call conprt(cls)
ccol=1
crow=1
def conprt(msg$)
                                                     fend
   for qi%=1 to len(msg$)
call conout(mid$(msg$, qi%, 1))
                                                     \=========FMTDATE$
    next qi%
                                                     def fmtdate$(da$)
fend
                                                         fmtdate$=mid$(da$,5,2)+"/"+
mid$(da$,3,2)+"/"+
mid$(da$,1,2)
\========TERMINIT
def terminit
   call conprt(trmini)
                                                     def fmtnum$(de,1e%)
   tmp$="
 "+str$(de)
def highlight
                                                         fmtnum$=right$(tmp$,1e%)
    call conprt(fullintensity)
                                                     fend
fend
                                                                        ======FMTDOL$
                                                     def fmtdol$(de,le%)
def lowlight
                                                         dd=de/100.00
tmp$="
    call conprt(halfintensity)
                                                                              "+str$(dd)
fend
                                                         tmp$=" "+str$(dd)
if match(".",tmp$,1)=0 then
    tmp$=tmp$+".00"
if mid$(tmp$,len(tmp$)-1,1)="." then
    tmp$=tmp$+"0"
def reverse
   call conprt(revideo)
                                                         fmtdol$=right$(tmp$,le%)
                                                     fend
                              ========UNREVRS
                                                     \=========VALIDS
def unrevrs
    call conprt(normvideo)
fend
                                                     def valid$(q$,type%)
    valid$=""
                                                         if type%=alpha then
                                                             if (asc(q\$)>31 and asc(q\$)<127) then
def bell
    call conprt(bellchar)
                                                                 valid$=q$
                                                         if type%=numeric then
   if match(q$,"-.+0123456789",1)>0 then
     valid$=q$
fend
def gotoxy(x%,y%)
                                                         if type%=upper then
                                                             ype%=upper L....
if match(q$,
   "ABCDEFGHIJKLMNOPQRSTUVWXYZ"+
   " 0123456789[]{}|^`=+_()!@#$%^*-/.&,
    rem screen 1-80 x 1-24 - 0 gives prev. posn
    if x%<>0 then crow=x%
                                                               " 0123456789[]()| = +chr$(34),1)>0 then
    if y%>0
        then ccol=y% `
    else
                                                               valid$=q$
        if y% < 0
                                                             else
                                                               if match(q$,
   "abcdefghijklmnopqrstuvwxyz",1)>0
            then ccol=ccol-y%
    call conprt(xyleadin+chr$
     (crow+31)+chr$(ccol+31))
                                                                     then
                                                                     q=chr(asc(q)-32) : valid=q
                                                         if type%=onlyalpha then
                                                             match(q$, \\
"ABCDEFGHIJKLMNOPQRSTUVWXYZ 0123456789."\
def displ(x%,y%,v$)
    call gotoxy(x%,y%)
                                                              ,1)>0 then
                                                              valid$=q$
    for qī%=l to len(v$)
        call conout(mid$(v$,qi%,1))
                                                     fend
```

Standard I/O Library continued.

# CB80/86

```
1=0
def vinput$(x%,y%,n%,type%)
                                                                   goto readch
                                                                 wrong:
call bell
    integer 1,i
dim ch$(80)
     readline:
                                                                 readch.
     call lowlight
                                                                   q$=coninp$
     call gotoxy(x%,y%)
for i=1 to n%
    call conout(".")
                                                                   if q$=right.arrow then
if 1 < n% then q$=mid$(deflt$,1+1,1)
                                                                   else
    next i call highlight
                                                                   goto wrong
qq$=valid$(q$,type%)
if qq$<>"" then q$=qq$ :
    if 1 < n% then</pre>
     call gotoxy (x%,y%)
     goto readch
                                                                            1=1+1
                                                                            ch$ (1)=q$ :
call conout(q$) :
  wrong:
call bell
  readch:
                                                                            goto readch
     q$=coninp$
                                                                   if 1<n% and
    match(q$,return.key+up.arrow+
                                                                        down.arrow+tab.key+escape,1)=0
                                                                   then goto 100 v$=""
              ch$(1)=q$:
                                                                   for i=l to l
              call conout(q$) :
                                                                        v$=v$+ch$(i)
                                                                   next i
              goto readch
     if 1<n% and
                                                                   for i=l+l to n%
                                                                       call conout (" ")
       match(q$,return.key+up.arrow+
       down.arrow+tab.key+escape,1)=0
                                                                   next i
                                                                   call gotoxy(x%,y%)
terminator$=q$
    then 100
    for i=l to l
                                                                   vupd$=v$
                                                                   return
         v=v+ch$(i)
    next i
                                                                100 if (q$=del.key or q$=home.key)
    for i=1+1 to n%
                                                                        then goto readline
         call conout(" ")
                                                                     if not (q$=back.space or q$=left.arrow) \
                                                                   then goto wrong
if 1 > 0 then
1=1-1:
    next i
    call gotoxy(x%,y%)
terminator$=q$
                                                                        call lowlight :
call conout (back.space) :
call conout (mid$(deflt$,l+1,i)) :
call conout (back.space) :
    vinput$=v$
     return
 call highlight
           then goto wrong
                                                                   goto readch
    if 1 > 0 then
                                                               fend
         1 = 1 - 1
         call lowlight:
                                                               \***************LUPD$
        call conout (back.space) : call conout (".") : call conout (back.space) : call highlight
                                                               def lupd$(x%,y%,deflt$)
    v$=vupd$(x%,y%,deflt$,alpha)
    if len(v$)=0
                                                                        then v$=def1t$
                                                                   else
    goto readch
                                                                        if len(v$)<>len(deflt$)
                                                                             then v$=v$+right$
                                                                             (deflt$,len(deflt$)-len(v$))
\===========LINPUT$
def linput$(x%,y%,n%)
    v$=vinput$(x%,y%,n%,alpha)
                                                                   call highlight
                                                                   call_displ(x%,y%,v$)
                                                                   1upd$=v$
    linput$=v$ ,
                                                               fend
                                                                                                   =====SYSERR
def vupd$(x%,y%,deflt$,type%)
  integer 1,i
  dim ch$(80)
  n%=len(deflt$)
                                                               def syserr(x%,y%,v$)
    call displ(x%,y%,v$+" (R)")
                                                                   call clreol ccol=ccol-2
                                                                   i$=linput$(0,0,0)
    readline:
    call gotoxy(x%,y%) call lowlight
    for i=l to n%
                                                               call conout(mid$(deflt$,i,1))
                                                               def yesno%(x%,y%,v$,d%)
    next i call highlight
                                                                   integer i,savcol
call displ(x%,y%,v$)
                                                                   savcol=ccol
    call gotoxy (x%,y%)
```

```
call disply1(0, 0, " Y/N")
                                                                        revdate$=
    if d%=yes then
call gotoxy(0,ccol-3)
                                                                       right$(vddmmyy$,2) +
mid$(vddmmyy$,3,2) +
left$(vddmmyy$,2)
    else
         if d%=no then
              call gotoxy(0,ccol-1)
                                                                                             ======NINPUT
         else
                                                                   def ninput(x%,y%,fln%,minval,maxval,defval)
              call gotoxy(0,ccol-2)
                                                                        n%=fln%
    goto getans
                                                                        if n%=0 then n%=log(maxval)+1
  wrong:
call bell
                                                                     getin:
                                                                       v\$=vinput\$(x\%, y\%, n\%, numeric)
if len(v\$)=0 then
  getans:
    result=defval :
                                                                             goto printit
                                                                        result=0
         then goto wrong
                                                                        for qi%= 1 to len(v$)
    if i=1 then goto nno
if i=2 then goto nno
if i=3 then goto yyes
                                                                            if i=4 then goto yyes
rem since i=5 ...
                                                                       next qi%
if (result<minval) or (result>maxval)
     if d%=no then goto nno
    if d%=yes then goto yyes rem since d=none or other...
                                                                             then goto getin
                                                                        printit:
                                                                       call displ (x%, y%, fmtnum$(result, n%))
call gotoxy (x%, y%)
ninput=result
    goto wrong
    yyes:
       call displ(0,savcol," YES")
                                                                   fend
       yesno%=true
       call displ(0, savcol, "NO ")
                                                                   def nupd(x%,y%,fln%,minval,maxval,defval,deflt)
       yesno%=false
                                                                        if n%=0 then n%=log(maxval)+1
fend
                                                                        vdef1t$=fmtnum$(def1t, n%)
                                                                       vdeflt$=imtnum$(deflt, n%)
for qi%=1 to n%
if mid$(vdeflt$, qi%, 1)=" " then
    vdeflt$=left$(vdeflt$,qi%-1)+"0"+
    right$(vdeflt$,len(vdeflt$)-qi%)
                                  ===============================VERIFY%
def verify%(tmpstr$,patrn$)
     for qi%=1 to len(tmpstr$)
  if match(mid$(tmpstr$,qi%,1),patrn$,1)=0 \
                                                                        next qi%
    then t%=t%+l
next qi%
verify%=t%
                                                                     getin:
                                                                       v$=vupd$(x%, y%, vdef1t$, numeric)
if len(v$)=0 then
                                                                             result=defval :
                                                                       goto printit
result=0
 for qi%=1 to len(v$)
q%=match(mid$(v$,qi%,1),
"0123456789",1)
result=result * 10+q%-1
def vdate%(vddmmyy$)
     integer dd, mm, yy integer days(1)
     dim ďays(lĺ)
                                                                        next qi%
     for qi%=1 to 12
                                                                        if (result < minval) or (result > maxval)
         days(qi%)=31
     next qi% days(4)=30
                                                                             then goto getin
                                                                     printit:
                                                                       call displ (x%, y%, call gotoxy (x%, y%) nupd=result
                                                                                                fmtnum$(result, n%))
     days(6)=30
     days(9)=30
     days(11)=30
     if verify%(vddmmyy$, "0123456789")<>0
          then vdate%=false : return
     dd=val(left$(vddmmyy$,2))
mm=val(mid$(vddmmyy$,3,2))
yy=val(right$(vddmmyy$,2))
                                                                   def doscan%(patrn$,base$)
                                                                        count%=0
                                                                        for qi%=1 to len(base$)
if match(patrn$,mid$(base$,qi%,1),1)>0 \
     if mod (yy,4)=0
then days (2)=29
                                           \ leap year
                                                                             then count%=count%+1
                                                                        next qi%
          days (2)=28
     if (dd<1 or mm<1 or mm>12 or
    dd>days(mm)) then
    vdate%=false:return
                                                                        doscan%=count%
     vdate%=true
                                                                    def isvalid$(fname$)
                                                                        while lefts(fname$,1)=" "
fname$=right$(fname$,len(fname$)-1)
 def revdate$(vddmmyy$)
```

```
if fname$="" then goto badn
     for x%=1 to len(fname$)
        if (asc(mid$(fname$,x%,1)) < 33 or
match(mid$(fname$,x%,1),"*,;<=>?[]",1)>0) \
           then goto badn
     next x%
     next x%
tmp%=match(":",fname$,1)
if tmp%=1 or tmp%>2 then goto badn
tmp%=match(".",fname$,1)
if (tmp%<>0 and tmp%<(len(fname$)-3))</pre>
     then goto badn
if doscan%(".",fname$)>1 then goto badn
if doscan%(":",fname$)>1 then goto badn
isvalid$=ucase$(fname$)
     return
   badn:
     isvalid$=""
fend
 def chrconv$(v$)
     rem backslash is escape char in match v\$=left\$(v\$,match("\\",v\$,1)-1) tmp\$=""
     tmps=""
while match(",",v$,1)
tmps=tmp$+chr$(val(left$(v$,
    match(",",v$,1)-1)))
v$=right$(v$,len(v$)-match(",",v$,1))
     tmp$=tmp$+chr$(val(v$))
     chrconv$=tmp$
fend
def get.parms
              screen control codes file \
     (STDIO.TRM) exists, use it
termfile$="STDIO.TRM"
     term%=7
tmp$=" "
     dim termch$(12)
     if size(termfile$)=0 then return
     if end # term% then errl
     open termfile$ as term% while left$(tmp$,2)<>"\\"
          read # term%; line tmp$
     wend
     for qi%=1 to 12
          read # term%; line termch$(qi%)
     next qi%
     cls=chrconv$(termch$(1))
     trmini=chrconv$(termch$(2))
     fullintensity=chrconv$(termch$(3))
     halfintensity=chrconv$(termch$(4))
     normvideo=chrconv$(termch$(5))
revideo=chrconv$(termch$(6))
     up.arrow=chrconv$(termch$(7))
down.arrow=chrconv$(termch$(8))
     left.arrow=chrconv$(termch$(9))
     right.arrow=chrconv$(termch$(10))
```

```
eraeol=chrconv$(termch$(11))
    xyleadin=chrconv$(termch$(12))
    close term%
    return
 errl:
      close term%
      print:print "Badly constructed ":termfile$
      stop
fend
def trim$(v$)
    while right $(v$,1)=" "
        v$=left$(v$,len(v$)-1)
    wend
    trim$=v$
fend
\ =======
def delay(length)
    qi=(length*dlyfactor)
    for waitloop=1 to qi
         x=abs(10)
    next waitloop
fend
                     =====ERR.HANDLER
def err.handler
a$=err : i%=errl
    v$="Oops. Program error! Code="+
a$+" line="+chr$(i%+48)
    call syserr(23,1,v$)
    if a$="EX" then call syserr(24,1,
"MP/M extended error - code"
+chr$(i%+48))
    stop
fend
```

# 

#### SCISYS SERIAL PRINTSHARE

An intelligent 4-channel switch device with built-in (64k) buffer — flexible configuration — copy function, LED display & lots more features.

PRICE — \$450 + sales tax



539 PITTWATER RD., BROOKVALE 2100. (02) 93-1383, (02) 908-1718

#### dBase

#### FIXDBF.BAS — The dBase User's Lifesaver

dBase can be a real pain in the neck, and not just because of its limitations — it's great to use, but when you lose a database file or its header is damaged you can be in real trouble.

Yes, it should be backed up ... but in the real world that often doesn't happen. When a file goes, you just have to grit your teeth and start typing again.

A number of programs are around which let you fix the record count in the dBase file header, but they solve only one of the problems. There are others—like the possibility of control characters being placed in the file, an end-of-file marker (control-Z) ending up in the wrong place or, worst of all, a power failure 'destroying' tens or hundreds of records because the buffers weren't written to disk and the header updated.

The last of these is the worst; if it's happened to you, take heart — FIXDBF has recovered hundreds of lost records for us at *Your Computer* in exactly the same situation in the past. And it can fix the other problems, too.

FIXDBF reads a database file a byte at a time and writes it out to a new file — either another database file or, if the header is completely garbled, it can re-

move the header and write out a text file which can then be appended to a new database.

It can be invoked with the two (old and new) filenames on the command line; otherwise it will prompt for the names. It then asks several questions: whether you want to produce a database file or a text file; whether you want to replace the record count (if you do, it then asks whether you want to provide the new count yourself; otherwise it 'measures' the file, and supplies its own to suit); whether you want to filter control-Zs in the body of the file; and whether you want to filter other control characters in the file. If you say yes to filtering control characters it asks whether you want them replaced automatically (with a space) or whether it should ask you each time.

It then lists the database structure (so you can tell if it is garbled) and asks whether the record size contained in the structure is correct — if not, you can nominate the record size yourself.

It's been a life-saver for us many times — hopefully it can do the same for you if you're unlucky enough to need it.

Matt Whelan

If you want to save yourself some typing, or don't have a CBASIC Compiler, you can get a compiled version of FIX-DBF on single-sided CP/M 20 cm diskette or IBM PC 13 cm diskette by sending a cheque or money order for \$25 to MotorWord Pty Ltd. 5 Clifton Road. Clovelly 2031. Compiled versions are available for CP/M-80. CP/M-86 and PC-DOS — please specify which operating system with your order.

```
%include stdio.dcl rem - declare globals
integer recsize, ccount, zcount, actual.recs
integer textfile, clean. finish, dbasefile integer scan, found, dummies, zfilter, cpmeof
         cfilter, fixhdr, disphdr
integer
integer last.cpmeof,askhdr
         in.file.name,out.file.name
string
string
         cr, nullname, crlf, vers, revision
%include stdio.ass rem - globals assignments
                          : crlf=chr$(13)+chr$(10)
: revision = "0"
cr=chr$(13)
vers = "2"
                            revision =
cpmeof = 26
                            control\% = 31
maxasc%
                            dbasefile =
           = 127
infile%
                            outfile% = 2
count% = 0
                            recount% = 0
working%=50
                            newline%=0
zfilter=false
fixhdr=false
                            cfilter = false
                            disphdr=false
dbheader%=521
                    rem dbase 521-byte header
zcount=0
                          : ccount=0
ask.replace=false
                            askhdr=false
clean.finish=false
                            textfile=false
coloffset%=15
copyrecs%=10000 rem initialise to lge no. recs nullname=string$(10,chr$(0)) rem empty dbase fld
%include stdio.bas
                       rem - i/o and misc functions
on error goto errhandler
call get.parms
                        rem - get terminal control
                         rem - codes if they exist
call clrscr call disply1(5,25,
  "dBase Data File Fixer "+vers+"."+revision)
call disply1(6,20,
   "Matt Whelan, Your Computer magazine")
\see if filenames specified on command line,
\else get file names
a$=command$
if a$="" then
    gosub getname
else
gosub breakname rem check which operations are wanted
gosub getchoices
gosub namecheck
dim header.array%(dbheader%)
     if end # infile% then finito
if end # outfile% then ofile.error
open in.file.name as infile%
     create out.file.name as outfile%
     gosub read.header
     if textfile then zfilter=true:
        cfilter=true:ask.replace=false
     if not textfile then
        gosub display.header
     if askhdr then
        gosub ask.header
        if fixhdr then
            gosub fix.header
```

if not textfile then

gosub copy.database

finito:

gosub write.header

FIXDBF.BAS continued.

# dBase

```
if not yesno%(crow+1,coloffset%,
   "Use physical record count (Y),"+
   " or ask? (N) ",none)
       if clean.finish
            then gosub fini2
                                                                                                            then askhdr=true
            gosub finished
                                                                                                     if yesno%(crow+1,coloffset%,
"Filter Control-Zs in the body "+
"of the file? ",yes)
then zfilter=true
       if not textfile then gosub print.stats
       stop
                                                                                                     if yesno%(crow+1,coloffset%,
    "Filter other control characters"+
    " in file ? ",yes)
getname
      call displyl(crow+3,20,
"dBase file to read from? ")
                                                                                                            then cfilter=true
       fname$=vinput$(0,0,14,alpha)
in.file.name=isvalid$(fname$)
                                                                                                     if (zfilter or cfilter) then
  if yesno%(crow+1,coloffset%,
          "Ask before replacing control"+
          "characters ? ",no)
       call displyl(crow+1,20,

"New file to write to ?
fname$=vinput$(0,0,14,alpha)
out.file.name=isvalid$(fname$)
                                                                                                                    then ask.replace=true
                                                                                                     return
       return
                                                                                              read.header:
breakname:
       m%=match(" ",a$,1)
                                                                                                     rem dbase 521-byte header
                                                                                                     for d%=1 to dbheader%
       if m%=0 then
              if ucase$(a$)="HELP" then
                                                                                                            header.array%(d%)=get(infile%)
                     goto helpinfo
                                                                                                     recs%=header.array%(3)*256+header.array%(2)
recsize=header.array%(8)*256+header.array%(7)
              else
       goto usage in.file.name=isvalid$(left$(a$,m%-1))
                                                                                                     return
       out.file.name=isvalid$(right$(a$,len(a$)-m%))
                                                                                              display.header:
                                                                                                     print
                                                                                                      if header.array%(1)<>dbasefile then
                                                                                                            call displyl(23,1,
"File is NOT marked as a "+
"dBase II database") :
       if in.file.name="" or
          out.file.name="" then
                                                                                                    "File 1s No."

"dBase II database"):
call syserr(24,1,
"Program behaviour may "+
"be 'unpredictable'!"):
call disply1(23,1," "):
call disply1(24,1," ")
print:print tab(coloffset%);

"Total records "; recs%
print tab(coloffset%); "Last update
str$(header.array%(4));"/";
str$(header.array%(5));"/";
str$(header.array%(6))
print tab(coloffset%); "Record size
       goto usage
m%=match(".",in.file.name,l)
if m%=0 then
             in.file.name=in.file.name+".DBF"
       if size(in.file.name)<1 then
  call syserr(23,1,
  "Can't find file "+in.file.name)</pre>
       : print : goto usage
m%=match(".",out.file.name,1)
if m%=0 then
              if not textfile then
                    out.file.name=
                    out.file.name+".DBF"
                                                                                                    print tab(coloffset%); recsize; bytes"
                                                                                                     pos%=8
                    out.file.name=
                    out.file.name+".TXT"
                                                                                                     Size"
        if size(out.file.name)>0 then
                                                                                                     for n%=1 to 32
              print:print:
                                                                                                         fieldname$=""
              if not yesno%(23,1,
"Output file exists...overwrite?
,yes) then stop
                                                                                                         for o%=1 to 10
if header.array%(pos%+o%)<>0 then
                                                                                                            fieldname$=fieldname$+
                   call disply1(23,1," ")
                                                                                                            chr$(header.array%(pos%+o%))
        if in.file.name=out.file.name then
                                                                                                        next o%
                                                                                                        next o%
spacing$=string$(15-len(fieldname$)," ")
if (fieldname$<>"" and fieldname$<>cr)\
    then print tab(coloffset%);
    fieldname$; spacing$;
    chr$(header.array%(pos%+12));
    " ";header.array%(pos%+13)
              call syserr(23,1,
"Sorry, you can't write the"+
" file to itself"):
              stop
       return
getchoices:
       if not yesno%(crow+3,coloffset%,
    "Copy database(Y), or produce "+
    "text file(N)? ",yes) then
    textfile=true : return
                                                                                                            n%=33
                                                                                                            pos%=pos%+16
                                                                                                    next n%
       textfile=true : return
if yesno%(crow+1,coloffset%,
    "Replace record count ?"+
    ",yes)
                                                                                                    print
                                                                                             check.ok:
                                                                                                   rem - routine to allow user to input n
rem - header info to go here
rem - (in case of complete corruption)
                                                                                                               routine to allow user to input new
               then fixhdr=true
       if fixhdr then
                                                                                                     rem - for the moment we will just check
```

# dBase

FIXDBF.BAS continued.

```
rem - whether recsize is ok
if not yesno%(23,coloffset%,
   "Is the record size - "
   +str$(recsize)+" bytes - okay?
                                                                               if not ask.replace then
                                                                               a%(x%)=32 : goto 100
print:print : call displyl(24,1,  \
"Found a control character in record "\
                                                                                  +str$(recount%))
        , none) then
"Actual record size?
                                                                               for i%=l to x%
                                                                                 pr 1%=1 to x%
if a%(i%)>control% and a%(i%)<maxasc% \
    then print chr$(a%(i%));
if a%(i%)<=control% then print " ^"; \
    chr$(a%(i%)+64);" ";
if a%(i%)>maxasc% then print " ^"; \
    chr$(a%(i%)-64);" ";
        ;recsize
 return
ask.header:
     print tab(coloffset%);:
                                                                               next i%
      input
                                                              \
"; xx%
"New record count?
                                                                               print
     header.array%(2)=mod(xx%,256)
header.array%(3)=int%(xx%/256)
                                                                               if yesno%(24,1,"Replace it ",yes) then \
                                                                                    gosub asknum
     copyrecs%=xx%
                                                                         100 print : print tab(coloffset%);
     return
                                                                               return
write.header:
                                                                         asknum:
     rem dbase 521-byte header
                                                                               print
                                                                               input "Replace with what? (decimal) ";new%
     for d%=1 to dbheader%
     put outfile%,header.array%(d%)
next d%
                                                                               a%(x%)=new%
                                                                               return
     return
                                                                         fix.header.
copy.database:
                                                                               rem - we have to read through the file
                                                                               rem - once to determine size!
     dim a%(recsize)
     last.cpmeof=false
                                                                               rem - a calculation based on filesize
rem - is inaccurate . . .
     print : print : print
print tab(coloffset%-9);"working ";
while recount%<copyrecs%</pre>
                                                                               print:print:print
                                                                               print tab(coloffset%);
                                                                               "Analysing data file...please wait "
if end # infile% then restart
        for x%=1 to recsize
           a%(x%)=get(infile%)
if cfilter then
  if (a%(x%)<=control% or</pre>
                                                                               eof.markers%=0
                                                                              while true
for r%=1 to recsize
                 a\%(x\%)>maxasc\%) and
                                                                                    q%=get(infile%)
                 a\%(x\%) <> cpmeof
                                                                                    if q%=cpmeof then
                 then gosub goask
                                                                                      eof.markers%=eof.markers%+1
           if zfilter then
if a%(x%)=cpmeof then
gosub check.eof
                                                                                 next r%
                                                                                 actual.recs=actual.recs+1
                                                                              wend
              elše
                                                                         restart:
                 last.cpmeof=false
        next x%
                                                                              actual.recs=actual.recs-
        recount%=recount%+1 : print "*"; :
                                                                                 (eof.markers%/recsize)
                                                                              copyrecs%=actual.recs
rem - close file after dummy read
        newline%=newline%+l
        if newline%=working% then print:
            print tab(coloffset%);
: newline%=0
                                                                              rem - then re-open and re-position for
                                                                              rem - actual processing close infile%
         for x%=1 to recsize
                                                                              open in.file.name as infile% if end # infile% then finito for d%=1 to dbheader%
q%=get(infile%)
           put outfile%,a%(x%)
        next x%
rem - add cr/lf for text files
        if textfile then put outfile%,13:
                                                                              next d%
                                                                              header.array%(2)=mod(actual.recs,256)
           put outfile%,10
                                                                              header.array%(3)=int%(actual.recs/256)
      wend
      clean.finish=true
      return
                                                                         print.stats:
                                                                              print:print:print
call disply1(23,coloffset%,
    "Input File: "+in.file.name)
check.eof:
     if last.cpmeof then
           goto finito
                                                                              print:print
      else
                                                                              call disply1(23,coloffset%,str$(recs%)+
           last.cpmeof=true: gosub goask
                                                                                   was the record count in file header")
      return
                                                                              print:print
                                                                              if not askhdr then
  call disply1(23,coloffset%,
  str$(recount%)+
  " was the number of physica
goask
      if a\%(x\%)=cpmeof then
           zcount=zcount+1
                                                                                   was the number of physical records")
      else
           ccount=ccount+1
```

FIXDBF.BAS continued.

#### dBase

```
call disply1(23,coloffset%,
    str$(copyrecs%)+" records c
if zfilter then
                                                                       helpinfo:
                                   records copied")
       if zcount>1 then print : print call disply1(23,coloffset%, str$(zcount-1)+
                                          print :
                                                                       usage:
                                                                       print
             unwanted control-Zs were found")
        if ccount>0 then print : print : call disply1(23,coloffset%, str$(ccount)+
     if cfilter then
                                                                       print
             control characters (not "+
           "including ^2) were found")
     print:print
     call disply1(23,coloffset%, "Output File: "+
          "Output File: "+
out.file.name) : print
     if fixhdr then print: print:
       print tab(coloffset%);
"**WARNING**": print tab(coloffset%);
        "Record count has been rewritten,
                                                                                          d:
        print tab(coloffset%);
        "- there may be garbage records" : print tab(coloffset%);
        "at end of file." : print : print tab(coloffset%);
         The new file should be PACKed":
                                                                       stop
        print
     return
                                                                       ofile.error:
                                                                            print
finished:
     for i%=1 to x%
     put outfile%,a%(i%)
next i%
     if textfile then put outfile%,13:
put outfile%,10
                                                                             stop
                                                                       errhandler:
fini2.
     put outfile%,cpmeof
                                                                            stop
     close infile%, outfile%
     return
```

```
helpinfo:
    print:print "Help messages to be put here!!"
    stop

usage:
print
print tab(coloffset%); "Usage:"
print tab(coloffset%); "FIXDBF, or"
print tab(coloffset%); "FIXDBF, or"
print tab(coloffset%); "where:"
print tab(coloffset%); "where:"
print tab(coloffset%); "(* and ? not allowed)"
print tab(
```

#### **FIND**

This program was created using the dBase II MODIFY command. dBase II lacks a command to search for duplicate records, but this program will search a database file and locate them. In my example the search is on a field called 'code' (this field must alread) be sorted) but it could be on any field.

Peter Grosvenor Yallambie, Vic

```
* FIND.CMD ......BY PETER.GROSVENOR. 20/1/85
* Find duplicate records based on the code
* save variables and restore them when done
SAVE TO temp
* start at the beginning of the file
GOTO TOP
* set up a loop to repeat until the whole file has been processed
* or the user decides to quit
STORE t TO more
DO WHILE more .AND. (.NOT. EOF)
* display something for the user to read
* while the program is searching for duplicates
ERASE
?
?
?
```

# dBase

```
am Searching for duplicate records Master
STORE !(code) TO oldcode
SKIP
IF oldcode = !(code) .AND. (.NOT. EOF)
? "please note duplicate code, and press any key to CONTINUE"
SET CONSOLE OFF
WAIT
SET CONSOLE ON
ENDIE
ENDDO
              I have finished Master
RESTORE FROM temp
```

For:

troducing

16 bit system at an affordable price.



Computer Pty. Ltd.

Suite 23 — Minton House, 2A Bayswater Road,

KINGS CROSS, NSW 2011

Tel: (02) 356-2388/356-2962

All correspondence to: P.O. Box 896, Potts Point, NSW 2011 

**Integrated Accounting Word Processing Spreadsheet** Point-of-Sale Cash Register Hookup

- Twin Floppy Double Density Drives
- Soft disk for 3rd drive
- 256K Expandable to 640K
- Multifunction card
- Colour/Mono Graphics OPTIONAL — 10/20 Megabyte Hard Disk

We also carry a wide range of printers, stationery, manuals, accessories, screens, etc.

## Commodore 64

#### DIAMOND MINER

This program lets you play Diamond Dan, a diamond miner. Your boss wants you to mine for as many diamonds as possible, and to speed you up a bit he has imposed a time limit within which to collect individual stones. The limit starts at 14 seconds, but if you get too fast—picking up a diamond in, say, five seconds—the boss may lower your time limit.

You must also be careful to collect only the white diamonds; the green ones are booby-trapped and will cost you a life. Watch out for rocks, as they will also reduce your life expectancy. The worst of all dangers are the walls: bumping into them will kill you instantly.

Added to all these dangers are red crosses which, when landed on, will move you randomly to another area of the mine.

With this game you can plug the joystick into either port. If you don't have a joystick, use the following keys instead:

A = left
D = r'ght
W = up
X = down

The codes listed below substitute for graphics characters on the Commodore.

(C/RT) = Cursor Right (C/DN) = Cursor Down (RVON) = Reverse On (RVOF) = Reverse Off (PURP) = Purple (LRED) = Light-red (LGRN) = Light-green (LBLU) = Light-blue (YELO) = Yellow (CLR) = Clear (HOME) = Home

Paul Vandenberg Cabramatta NSW

```
Ø REM DIAMOND MINER BY PAUL VANDENBERG
1 POKE5328Ø, Ø: POKE53281, Ø: PRINT " (PURP) "
  CH=5:1 Y=20:1 X=20
3 TI$="ØØØØØØ":TL=9ØØ
4 Ds="(C/DN)":Rs="(C/RT)"
5 JS=56320:GOSUB2000
6 S=54272:FORZ=STOS+24:POKEZ,Ø:NEXT:POKE
S+5,9:POKES+6,Ø:POKES+24,15
100 PRINT"(CLR)":
1Ø1 PRINT"(RVON)
102 FORF=1T020:PRINT" "; SPC(38); " ";:NEX
103 PRINT"
          {RVOF}"
1Ø5 CL=1:FORF=1T05:X=INT(RND(Ø) *38)+1:Y=
INT (RND(Ø) *2Ø) +1:C=9Ø:GOSUB1ØØØØ
1Ø6 NEXTF
109 FORF=1T015:A=PEEK(197):J1=PEEK(JS):J
2=PEEK(JS+1):POKES+4,32
11Ø POKES+1,5Ø:POKES+4,129:POKES+4,Ø:IFT
I>=TLTHENGOSUB8ØØØ
111 CL=0:C=32:X=LX:Y=LY:GOSUB10000:IFA=9
ORJ1=1260RJ2=254THENLY=LY-1
112 IFA=230RJ1=1250RJ2=253THENLY=LY+1
113 IFA=1Ø0RJ1=1230RJ2=251THENLX=LX-1
114 IFA=180RJ1=1190RJ2=247THENLX=LX+1
115 P=LY*4Ø+LX:L=PEEK(55296+P)
116 POKES+4,32: IFL=4THEN9000
117 IFL=6THENCH=CH-1:POKES+1.8:POKES+4.3
3: FORZ=1TO5Ø: NEXTZ: IFCH=ØTHEN9ØØØ
118 IFL=1THENPT=PT+100:CT=TI:Tİ$="000000
":GOSUB5000:IFCT>360THENTL=TL-20
119 IFL=5THENCH=CH-1:POKES+1,8:POKES+4,3
3:FORZ=1TO5Ø:NEXTZ:IFCH=ØTHEN9ØØØ
12Ø IFL=1ØTHENGOSUB5Ø5Ø:LX=INT(RND(Ø) #38
)+1:LY=INT(RND(Ø) *2Ø)+1
121 CL=14:C=42:X=LX:Y=LY:GOSUB10000
122 IFTI$="ØØØØ15"THENGOSUB8ØØØ
124 PRINT"(HOME)(RVON)(PURP)"TI$; TAB(10)
; "SCORE"; PT; TAB(26); "MEN LEFT"; CH
129 NEXTE
13Ø CL=1:FORF=1T01:X=INT(RND(Ø) *38)+1:Y=
INT (RND (Ø) *2Ø) +1: C=9Ø: GOSUB1ØØØØ
131 CL = 6: FORF = 1TO4: X = INT (RND (0) *38) +1: Y=
INT (RND(Ø) *2Ø) +1: C=224: GOSUB1ØØØØ: NEXTF
132 CL=5:X=INT(RND(Ø) *38) +1:Y=INT(RND(Ø)
*2Ø) +1:C=9Ø:GOSUB1ØØØØ
133 IFRND(Ø) >.8THENCL=10:X=INT(RND(Ø) *38
)+1:Y=INT(RND(Ø) *2Ø)+1:C=86:GOSUB1ØØØØ
16Ø GOT01Ø9
2000 REM TITLE PAGE
2001 PRINT"(CLR)";:FORZ=1T05:PRINTD$;:NE
XTZ
2002 PRINT"(PURP)";
2ØØ3 PRINT"(RVON) ₹(C/RT) {C/RT) ₹(C
/RT}¶(C/RT)(C/RT)F(C/RT)F ¶(C/RT)¶(C/RT)
(C/RT) (C/RT) ■"
2004 PRINT"(RVON) (C/RT) (C/RT)(C/RT) (C
/RT)(C/RT) (C/RT) (C/RT) (RVOF)\/(RVON)
(C/RT) (C/RT) (C/RT) (RVOF)\(RVON)(C/RT)
(C/RT) (C/RT) "
2005 PRINT"(RVON) (C/RT) (C/RT)(C/RT) (C
/RT)(C/RT) (C/RT)(C/RT) (C/RT)
(C/RT) (C/RT) (C/RT)(RVOF)\(RVON) (C/RT)
 (C/RT) "
2006 PRINT"(RVON) (C/RT) (C/RT)(C/RT) (C
/RT)(C/RT) (C/RT) (C/RT) (C/RT)(C/RT) (C
/RT) (C/RT) (C/RT) (C/RT) (C/RT) (
C/RT3 '
```

```
2007 PRINT"(RUON) (RUOF) CRUON)
                                     CC/R
T) (C/RT) (C/RT) (C/RT)(C/RT) (C/RT)(RVO
F) T(RVON) (RVOF) F(C/RT)(RVON) (RVOF) (C/
RT3(RVOE) ▼ (RVON) (RVOE) F'
2008 PRINT: PRINT
2009 PRINTTAR(10): "{RVON} \(C/RT){C/RT}\()
2010 PRINTTAB(10): "(RVON) (RVOF) \/ (RVON)
 {C/RT}{C/RT} {C/RT}{C/RT} {RV0F}\(RV0N)
(C/RT) (C/RT) (C/RT)(C/RT)(C/RT) (C/RT)
2011 PRINTTAR(10): "(RUON) (C/RT)(C/RT) (
C/RT)(C/RT) (C/RT)(C/RT) (C/RT)(RVOF)\(R
VON) (C/RT) (C/RT)(C/RT) (RVOF) [F" 2012 PRINTTAB(10); "(RVON) (C/RT)(C/RT) (
C/RT3(C/RT) (C/RT)(C/RT) (C/RT)(C/RT) (C
/RT) (C/RT)(C/RT)(C/RT) (RVOF)\(RVON)"
2013 PRINTTAB(10): "(RVON) (C/RT)(C/RT) (
C/RT) (C/RT) (C/RT)(C/RT)(RVOF) (RVON
    (C/RT) (C/RT)(RVOF)\"
2015 PRINT: PRINT" (C/RT) (C/RT) (C/RT)
ACCURTAGE GRNARY PAUL VANDENBERG"
2016 PRINT: PRINT" (LRED) PRESS A KEY TO ST
ART (PURP) "
2017 GETAS: IEAS=""THEN2017
2013 RETURN
5000 POKES+1,50:POKES+4,17:POKES+4,16:FO
RZ=1T02Ø:NEXTZ
5001 POKES+1,60:POKES+4,17:POKES+4,16:FO
RZ=1TO2Ø:NEXTZ
5002 RETURN
5050 FORF=40T070STEP2:POKES+1,F:POKES+4,
33:NEXTF:POKES+4.32:RETURN
7999 REM NEXT LEVEL MARKER""
8000 REM TIME UP
8001 PRINT"(HOME)";:FORZ=1TO23:PRINTD$;:
NEXTZ:PRINT"(PURP)YOU WERE A BIT SLOW"
8002 POKES+1,6:POKES+4,33:FORZ=1T02000:N
EXTZ: CH=CH-1: IFCH=ØTHEN9ØØØ
8003 PRINT" (C/UP)
                                      ( HOM
E)":POKES+4.32
8004 TI$="000000":RETURN
8500 RETURN
9000 PRINT"(CLR)YOU'RE DEAD!!"
9001 GETA$: IFA$<>""THEN9001
9002 PRINT" (LBLIDYOU'VE ACCUMULATED A TO
TAL SCORE OF
9ØØ3 PRINT:FORZ=1TO2Ø:PRINT"(C/RT)(C/RT)
(C/RT)(C/RT)(C/UP)(YELO)":PT:FORG=1TO40:
NEXTG: PRINT" (C/RT) (C/RT) (C/RT) (C/RT) (C/U
P)(GRN)";PT
9004 POKES+1.5:POKES+4.33:POKES+4.32
9005 FORG=1TO38:NEXTG:NEXTZ
9006 PRINT:PRINT"(LGRN)WOULD YOU LIKE AN
OTHER GO? (Y/N)"
9007 GETA$: IFA$= " "THEN9007
9008 IFAS="Y"THENCLR:GOTO1
9009 END
10000 P=Y*40+X:POKEP+1024,C:POKE55296+P,
CL:RETURN
```

#### Commodore 16

# COMPATIBILITY ANALYSIS

The program displays a compatibility table for two people based on the biorhythmic cycle theory. The table consists of physical cycle, sensitivity cycle and cognitive cycle compatibility.

Although it was written on a C16, the program should also run without modification on the Plus Four.

Mark Wilkinson Heufield VIC

```
10 REM *******************
                                COMMODORE 16
20 REM *
 30 REM * COMPATABILITY ANALYSIS *
40 REM * AUTHOR: M. WILKINSON
50 REM *****************
60 REM
70 DIMA1(30),B1(30),M$(12),D$(7),A(12)
80 COLOR0,3,2:COLOR4,6,3
90 FORI=1T012:READA(I):NEXT
100 DATA0,31,59,90,120,151,181,212,243,273,304,334
110 GOSUB800
120 Y=0:F0$="%###.##"
130 Y=Y+1
140 INPUT" INPUT I
150 INPUT"ŒBIRTHDAY (DD,MM,YYYY).∰";D,M,Y
160 E1=M:F1=D:G1=Y:GOSUB600
170 Z2=T:K1=J+1
180 INPUT"XXXXNAME OF PERSON 2..... 3#";X$
190 INPUT"EBIRTHDAY (DD,MM,YYYY).@";D,M,Y
200 E2=M:D2=D:G2=Y:G0SUB600
210 P2=ABS(Z2-T):K2=J+1
220 PRINT"3";
230 PRINTTAB(8)"3 ~
240 PRINTTAB(8)" DCOMPATABILITY ANALYSISCI"
250 PRINTTAB(8)"
260 PRINT TAB((40-LEN(W$))/2);W$
270 PRINT TAB((40-LEN(X$))/2);X$;"W"
280 PRINT
290 PRINT"%"W$":8"
300 J=K1:GOSUB790:PRINT",";
310 M=E1:GOSUB780
320 PRINTF1", "G1
330 PRINT"XX1"X$":22";
340 J=K2:GOSUB790:PRINT",";
350 M=E2:GOSUB780
360 PRINTD2", "G2"XXX"
370 Z≈P2
380 P3≃ABS(INT(((Z/23)-INT(Z/23))*23))
390 S3=ABS(INT(((Z/28)~INT(Z/28))*28))
400 C3=ABS(INT(((Z/33)-INT(Z/33))*33))
410 P5=ABS(100-((2*P3)*(100/23)))
420 S5=ABS(100+((2*S3)*(100/28)))
430 C5=ABS(100+((2*C3)*(100/33)))
440 PRINTTAB(9)"#
450 PRINTTAB(10)" #PHYSICAL.... (3")
460 PRINT USING FO$;P5
470 PRINTTAB(10)"編SENSITIVITY..篇";
480 PRINT USING FO$;S5
500 PRINT USING FO$;C5
510 PRINTTAB(9)"#
520 A5=(P5+S5+C5)/3
530 PRINTTAB(10)"#AVERAGE.....
540 PRINT USING FO$;A5
550 PRINTTAB(9)"#
560 PRINT" XXXXXX
570 PRINT"
                               PRESS ANY KEY FOR ANOTHER ANALYSIS =":
580 GETKEY A≸
590 PRINT"∭;:RUN
600 Y1=Y-1800
610 Q1=INT(Y1/4)
```

# Commodore

Compatibility Analysis continued

620 Q2=INT(Q1/25) 630 03=INT((Y1+200)/400) 640 K=0 650 IFQ1\*4<>Y1THEN690 660 IFQ2\*100<>Y1THEN690 670 IFQ3\*400-200<>Y1THEN690 680 K=1 690 T≈365\*Y1+Q1-Q2+Q3-K 700 T=T+A(M)+D-1 710 IFMCSTHEN730 720 T=T+K 730 IFINT(Y1/4)<>Y1/4THEN760 740 IFMD2THEN760 750 T=T-1 760 J=T-7\*INT(T/7) 770 RETURN 780 PRINTM\$(M); :RETURN 790 PRINTD\$(J);:RETURN 800 FORJ=1TO12:READM\$(J):NEXT 810 FORJ=1TO7:READD#(J):NEXT:RETURN 820 DATAJAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP 830 DATAOCT/NOV/DEC 840 DATAWEDNESDAY, THURSDAY, FRIDAY, SATURDAY

850 DATASUNDAY, MONDAY, TUESDAY 860 REM 870 REM 880 REM-----890 REM CONTROL CODES 900 REM IN THIS LISTING 910 REM 920 REM "3 = CLEAR 930 REM "X = DOWN 940 REM " = BLU 950 REM "# = PUR 960 REM "X = ORNG 970 REM "# = GRN 980 REM "# = FLASH ON 990 REM "■ ≈ FLASH OFF 1000 REM----

# BEEARTISTIC.

You have a choice of drawing tools: pencil, brush, spray-can, fill can, eraser.

Draw points, lines, boxes, elipses or handcrafted drawings.

User-definable patterns – defined areas may be filled or you may brush them on. Text and graphics can be combined; perfect for overhead projector artwork, etc.

"Cut & Paste" facility. Define your own icons and copy them.

Zoom editing – allows you to edit your art in fine detail.

Pictures can be saved on or retrieved from disk, or printed out.

BEESLIDE – a variable speed slide-show package; user defined order and files.

Min. Requirements: 32k Microbee, CP/M, 1 disk drive (40 track DSDD), Epson compatible printer (option).

THE ONLY
GRAPHICS PACKAGE
AVAILABLE FOR
THE
MICROBEE

DESIGNED AND DEVELOPED IN AUSTRALIA

# A FEW MENUS

I want to BEEARTISTIC too!
Cheque 
Bankcard 
Mastercard 
VISA 
BEEARTISTIC Packages 
\$49.95
Postage and Packaging \$3.50

 Postage and Packaging @ \$3.50\_\_\_\_\_\_

 NAME:
 \_\_\_\_\_\_EXP:
 \_\_\_\_\_\_\_

 ADDRESS:
 \_\_\_\_\_\_P/CODE

**EXITEK** 

P.O. Box 348, Balwyn North, Victoria. Australia. 3104. (03) 846 3500

# Microbee 🦝

#### **MOTORBIKE TRIALS**

Motorbike Trials simulates a rider travelling a twisting 100 kilometre course

The rider starts at the top of the screen facing down. The 'A' key moves him to the player's right and the '\' key moves him to the player's left.

Every few kilometres the rider moves toward the bottom of the screen and is followed by another rider (you always control the front one). If your rider reaches the bottom of the screen, you've completed the course — it's very hard!

Good luck ... and watch for slow-moving cars.

Ken Rowe Dernancourt SA

00820 RETURN

```
00090 GOTO 700
00095 CLEAR: RESTORE
00100 F=63488+49*16
00110 FOR A=P TO P+16#8-1
00120 READ B:POKE A,B
00130 NEXT A
00140 DATA 36,255,255,60,36,189,189,126,126,60,60,189,255,189,24
00170 DATA 0,16,16,16,40,56,124,108,186,68,56,16,16,16,0,0
00180 DATA 0,0,1,2,4,30,46,73,50,28,32,32,64,128,0,0
00190 DATA 0,0,128,64,32,32,124,114,73,42,20,4,2,1,0,0
00200 DATA 8,137,73,34,20,93,46,86,28,99,22,236,26,41,68,32
00210 DATA 15,15,15,15,240,240,240,240,15,15,15,15,240,240,240,240
00220 CLS: R=1:N=33:E=5
00230 GOSUB 520
00240 IF RND#1<.2 THEN LET R=INT(RND#3)
00250 K=K-1+R: Z=0
00280 IF PEEK(258)=28 THEN LET N=N+1:Z=2 ELSE IF PEEK(258)=1 THEN LET N=N-1:Z=1
00290 PCG:POKE 61440+(@#64)+N,180+Z:NORMAL
00300 IF K>20 THEN LET K=20 ELSE IF K<-29 THEN LET K=-29 00320 IF RND*2<.112 THEN 460
00330 PRINT TAB(30+K);:PCG:PRINT"22";:NORMAL:PRINT SPC(E);:PCG:PRINT"33":NORMAL
00340 IF PEEK(61440+(64*Q)+N)<180 AND PEEK(61440+(64*Q)+N)>175 THEN 370
00350 S1=S1+1: IF S1=100 THEN GOSUB 800
00360 GOTO 240
00370 POKE 61440+(64*Q)+N,183:FOR I=1 TO 22:IF RND*1<.55 THEN OUT 2,59:OUT 2,65:
NEXT I ELSE NEXT I
00380 PLAY 0,1;22,5;19,5;22,5;19,5;22,5;19,5
00390 F=@*100+INT(81):CURS1:PRINT"YOU COMPLETED ":INT(FLT(F)/1500*100):" Km OF A
100 Km COURSE."\"AND PICKED UP ANOTHER ";Q;" RIDERS"\"1] ANOTHER GAME"\"2] END"
00400 A1$=KEY$:IF A1$="" THEN 400
00410 IF A1$="1" THEN 95 ELSE IF A1$="2" THEN PRINT\"BYE":END
00420 GOTO 400
00430 PRINT TAB(30+K);:PCG:PRINT"22";:FDRI=1TDE:PRINT"8";:NEXTI:PRINT"33":NORMAL
00440 PLAY5,5;5,5;7,5;9,5;7,5;5,5;5.5:CURS1:PRINT"YOU HAVE COMPLETED THE 100 Km COURSE WITH 14 RIDERS"\"THAT IS AN EXCELLENT ACHIEVEMENT!"\"1] ANOTHER GAME"\"2]
 END": GOTO400
00450 END
00460 IF RND*1<.35 THEN 490
00470 W=INT(RND*(FLT(E))+1):PRINTTAB(30+K);:PCG:PRINT"22";:NORMAL:PRINT SPC(W-1)
;:PCG:PRINT"1";:NORMAL:PRINTSPC(E-W);:PCG:PRINT"33":NORMAL
00480 GOTO340
00490 IF RND*1<.5 THEN LET E=E+1 ELSE LET E=E-1
00500 IF E<4 THEN LET E=4 ELSE IF E>8 THEN LET E=8
00510 GOTO 330
00520 NORMAL: FOR I=1 TO 15:IF I=2THEN570 ELSE PRINT TAB(30);:PCG:PRINT"22"::NOR
MAL:PRINT SPC(5);:PCG:PRINT"33":NORMAL:NEXT I
00530 PDKE 61440+N.180
00535 CURS5,2:PRINT"<== 'A'";:CURS54,2:PRINT"'\' ==>";:CURS15,16
00540 FOR I=1 T05:FOR D=1 TO 25:OUT 2,59:OUT 2,65:FOR G=1 TO D
00550 NEXT G:NEXTD:NEXTI
00560 RETURN
00570 PRINT TAB(30);:PCG:PRINT"228888833":NORMAL:NEXT I
00700 CLS:PLAY 16,2;16,2;13,4;16,4;18,2;16,4;18,4;18,4;18,4;20,2;18,2;16,2;16,4
00710 INPUT " INSTRUCTIONS ? ";T1$
00720 IF T1$="YES"ORT1$="yes"ORT1$="y"ORT1$="Y"THEN730ELSE100
00730 CLS: PRINT"
                                               ** MOTORBIKE TRIALS **"\"
           ***********
00740 PRINT" You are set a difficult 100 km course."\"Along the way you are joi ned by more riders."\"You will lead these riders through the winding corners" 00750 PRINT"And past slower cars."\\"The direction of the motorbike is to the bottom of the screen"\"'A' moves the bike to your left & '\' to your right."
00760 PRINT\"The road changes width as well as direction."\\"If you can pick up another 14 riders and reach the bottom of "\"the screen, you have completed the
course (very hard!!)"
00770 INPUT
                                                           HIT <<RETURN>> TO START": T4$
00780 GOTO 100
00800 Q=Q+1: IF PEEK(258)=28THENLETN=N+1 ELSE IF PEEK(258)=1THENLETN=N-1
00810 S1=0 : IF Q=15 THEN 430
```

# Microbee 🐷

#### **CLOCK**

Here is a program for all the egg burners among you. Just set the clock and alarm for perfect eggs, or use the stop watch for timing those three-hour chess games. The letter 'S' will stop and start the stop watch; press 'R' after the watch is stopped to return to the menu

The program is quite easy to convert to other machines, given that 'Lores' and 'Hires' select graphics modes for the borders around the outside of the menu and the introduction. All 'Plots' can be left out, or your own computer's graphics commands may be substituted. CLS clears the screen. The Key\$ is the same as Inkey\$ on most other machines. The rest is fairly universal.

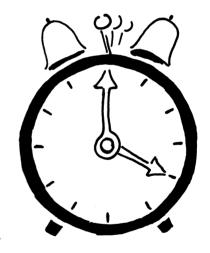
Jeremy Fenton Lismore NSW

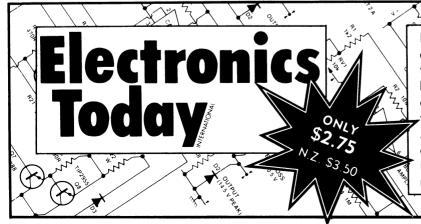
```
88118 REM ## CLOCK BY J. FENTON
66126 REN ## FOR PUBLIC USE
                                                H
88138 REM ## LAST UPBATE 2/3/85
99159 LET X=0
88178 CLS
99189 HIRES
88198 PLOT 9,8 TO 8,255 TO 511,255 TO 511,8 TO 8,8
88288 PLOT 198,158 TO 58,158 TO 58,288 TO 188,288
99219 PLOT 128,288 TO 128,159 TO 178,159
88228 PLOT 198,288 TO 198,158 TO 248,158 TO 248,288 TO 198,288
99239 PLOT 319,299 TO 269,299 TO 269,159 TO 319,159
96246 PLOT 336,266 TO 336,156:PLOT 336,175 TO 396,266: PLOT 331,176 TO 396,156
##25# CURS 4#,15:PRINT*PRESS ''RETURN''*
99269 A3$=KEY$:IF A3$="" THEN GOTO 269
99279 GOTO 728
88298 CLS
##3## CLS:CURS 21,7:INPUT"ENTER TIME>-HOURS"H
99319 CURS 32,8: INPUT"-MINUTES"M
##32# CURS 32,9:INPUT"-SECONDS"S
66336 RETURN
00350 CLS
99369 GOSUB 579
99379 GOSUB 629
99389 GOSUB 639
00390 GOSUB 640
99499 S=S+1
66416 FOR A6=1 TO 31 STEP.1687899
66426 NEXT A6
##43# GOSUB 64#
86448 IF S(68 THEN 488
99459 S=9
99469 GOSUB 649
66476 M=M+1
69488 GOSUB 638
88498 IF M(68 THEN 488
88588 H=8
99519 GOSUB 639
99529 H=H+1
88538 GOSUB 628
88548 IF H(13 THEN 488
99559 H=1
99569 GOTO 379
99579 CURS 20,7:PRINT HOURS MINUTES SECONDS"
```

# Microbee 🐷

88588 CURS 38.15:PRINT"PRESS "'R" FOR MENU" GGAGG RETIEN 66626 CURS 21.8:PRINT H" ":RETURN ##63# CURS 29,8:PRINT Nº ":RETURN \$6640 CURS 37.8:PRINT S" ": IF Q=H AND K=M AND V=S THEN PLAY 1;1;1;1;1;1;1;1;1;1;1;1 99659 A5\$=KEY\$: IF A5\$="R" OR A5\$="r" THEN GOTO 729 88668 RETURN ##68#CLS: CURS21.8:IMPUT\*HOUR >\*;@ \$6696 CURS 21,9:INPUT\*MINUTES >";K \$6786 CURS 21.18: INPUT SECONDS >";V 99718 RETURN 89728 CLS \$6746 LORES: PLOT \$,6 TO \$,47 TO 127,47 TO 127,5 TO \$,6 99759 CURS 39,5:PRINT"MENU" 66768 CURS 18,7: PRINT"SET THE CLOCK : S' ##77# CURS 1#,9: PRINT"STOP WATCH 88788 CURS 18,11:PRINT"SET THE ALARM : A' ##79# CURS 1#,13:PRINT'START CLOCK : C" 88888 A5\$=KEY\$: IF A5\$="" THEN GOTO 888 \$8618 IF A5\$="S" OR A5\$="s" THEN GOSUB 388 86826 IF A5\$="a" OR A5\$="A" THEN GOSUB 686 ##83# IF A5\$="C" OR A5\$="c" THEN GOTO 35# 88848 IF A5\$="W" OR A5\$="w" THEN GOTO 878 ##85# GOTO 72# 99869 REM TELEFFERENCE STOP WATCH THEFFERENCE THEFF 66876 CLS 99889 CURS 29,6:PRINT" MINUTES SECONDS ":GOTO 999 66896 FOR P=1 TO 866:NEXT P

88918 LET Z=Z+1
88928 CURS 34,8:PRINT Z
88938 CURS 25,8: PRINT X
88948 A5\$=KEY\$:IF A5\$="S" OR A5\$="s" THEN GOTO 978
88958 IF Z=68 THEN LET Z=8: X=X+1: GOTO 878
88968 GOTO 898
88978 A5\$=KEY\$:IF A5\$="" THEN GOTO 978
88988 IF A5\$="S" OR A5\$="s" THEN GOTO 958
88998 IF A5\$="R" OR A5\$="r" THEN LET Z=8:LET X=8:GOTO 728
91888 GOTO 978





99999 A5\$=KEY\$:IF A5\$="S" OR A5\$="s" THEN GOTO 978

Electronics Today is Australia's dynamic electronics monthly. It has more special features, new and exciting projects to build and a wealth of information on components, equipment and new technology. Regular features include Australia's top hi-fi reviews and news on communications and computing. Buy your copy now from your local newsagent.

#### **VZ200**

#### NUMBER SEQUENCE

This program prints various sequences of numbers, each ending with a blank. You must enter the next number in the sequence — the computer indicates if your entry was correct.

A series of ten questions is asked, then your score is given.

Because the program is written in standard Microsoft BASIC, it should be easily transported to other computers. The random number statements in lines 120-140 may need modification, according to your particular version of BASIC.

lan Thompson Collaroy Plateau NSW

```
NUMBER SEQUENCE
 * FOR THE UNEXPANDED VZ-200
 * IAN THOMPSON - COLLARDY
 *********
10 CLS:PRINT@104, "NUMBER SEQUENCE"
12 PRINT@325, "IAN THOMPSON, COLLAROY"
15 PRINT@485, "PRESS ANY KEY TO START"
20 IF INKEY$="" THEN 20
21 IF INKEY$="" THEN 20
25 CLS:PRINT"
                     NUMBER SEQUENCE": PRINT
30 PRINT"THIS PROGRAM WILL PRINT VARIOUS"
35 PRINT"SEQUENCES OF NUMBERS, EACH
40 PRINT"ENDING WITH A BLANK (----)."
45 PRINT"WHEN YOU SEE A '?', TYPE IN THE"
50 PRINT"NUMBER THAT YOU THINK THE "
55 PRINT"COMPUTER MIGHT HAVE PRINTED IN "
60 PRINT"PLACE OF THE BLANK."
70 PRINT
75 FRINT"**********************
80 LET R=0
90 LET W=0
100 FOR I=1 TO 10
110 PRINT"PROBLEM": I
120 LET A=INT(10*RND(0)+1)
130 LET B=INT(10*RND(0)+1)
140 LET G=RND(3)
150 IF A>B THEN 285
160 IF G=1 THEN 170
162 IF G=2 THEN 210
164 IF G=3 THEN 250
170 LET X=2*A+3*B
180 PRINT A; ", "; B; ", "; A+B; ", "; A+2*B; ", ----";
190 INPUT Y
200 GOTO 410
210 LET X=A*A*B*B*B
220 PRINT A; ", "; B; ", "; A*B; ", "; B*A*B; ", ----";
230 INPUT Y
240 GOTO 410
250 LET X=-B
260 PRINT A; ", "; B; ", "; B-A; ", "; -A; ", ----";
270 INPUT Y
280 GOTO 410
285 IF G=1 THEN 300
290 IF G=2 THEN 340
300 LET X=A*5
310 PRINT A; ", "; 2*A; ", "; 3*A; ", "; 4*A; ", ----";
320 INPUT Y
330 GOTO 410
340 LET X=16*A
350 PRINT A; ", "; 2*A; ", "; 4*A; ", "; 8*A; ", ----";
360 INPUT Y
410 IF X=Y THEN 450
420 PRINT"NO: THE COMPUTER'S SEQUENCE HAS ":X:"."
430 LET W=W+1
440 GOTO 470
450 PRINT"THAT'S RIGHT!"
460 LET R=R+1
470 PRINT
480 NEXT I
495 SOUND 15,5
500 PRINT"SCORE: ";R;" RIGHT,";W;" WRONG
505 PRINT:PRINT"===================
510 PRINT" PRESS (SPACE) FOR ANOTHER SET
520 PRINT" QUESTIONS."
530 A$=INKEY$:IF A$ <> " "THEN 530
535 RUN
```

#### TRS-80

#### **MAILIST**

Mailist is a mailing list program for TRS-80 Models I, III and IV with 16 Kbytes of RAM. It stores up to 250 names and addresses on cassette tape for later retrieval and printing.

10 ' MAILIST.

For those of you who have machines with 48 KBytes of RAM, a version for up to 1100 names and addresses with a built-in lowercase driver is available for \$10, cassette supplied.

Both programs are menu-driven and a help sequence is included in the program (function 8). If you have any questions or wish to purchase the 48 Kbyte version, write to David Minehan, 64 Young Road, Lambton East 2299

David Minehan Lambton East, NSW

```
20 ' PROGRAMMER: DAVID KEITH MINEHAN.
30 ' SYSTEM: TRS-80/SYSTEM-80.
40 ' RAM: 16K.
50 ' LANGUAGE: BASIC.
60 '
    PROGRAM DESIGNATION: BUSINESS.
70 '
80 '
    PROGRAM DESIGNED TO TAKE NAMES,
90 ' ADDRESSES, AND PHONE NUMBERS &
100 ' STORE THEM ON CASSETTE-TAPE
110 '
     DATA FILE(S).
120 '
220 CLEAR5500:CLS:POKE16396,23
   :DIMN#(250),R#(250),R#(250),P#(250)
230
   PRINT TAB(30); "MAILIST": PRINT: PRINT: ' ***MAIN MENU*** '
240
   PRINTTAB(10)"CREATE FILE(8).....1"
250 PRINTTAB(10)"VIEW FILE(S).....2"
260 PRINTTAB(10)"UPDATE FILE(S).....3"
270 PRINTTAB(10)"SAVE FILE(S)......4"
280 PRINTTAB(10)"CALL
                      290 PRINTTAB(10)"VERIFY FILE($).....6"
310 PRINTTAB(10)"HELP.....8"
320 PRINTTAB(10)"QUIT.....
330 PRINT:PRINTTABK 10 > "WHICH FUNCTION DO YOU WISH TO EXECUTE" :: INPUTQ
340 ONGGOTO410,510,610,710,810,910,1010,1110,9999
350 CLS:G0T0230:
                  / ***MENU END***
410 CLS:INPUT"WHEN READY, HIT (ENTER) (TO CLOSE FILE TYPE 9999 FOR NAME)";X:CLS
420 FORI=1T0250:PRINT:PRINT"ENTER NAME (LAST FIRST, NO COMMAS PLEASE)"
430 PRINT"THEN HIT THE (ENTER) KEY"; : INPUTN#(I)
440 IFN#(I)="9999"THENP1=I:GOTO490 '
445 PRINT"ENTER NAME OF REPRESENTITIVE ((ENTER) FOR NIL)"
446 INPUTR#(I)
450 INPUT"ENTER STREET NUMBER & NAME (NO COMMAS)"; A#(I)
460 INPUT"ENTER SUBURB/CITY & POSTCODE"; P#(I)
470 IFFRE(X)<250GOTO490:IFFRE(X)=250THENPRINT:PRINT"FILE FULL"
    PRINT
   :INPUT"PRESS (ENTER) TO RETURN TO MENU";X
    · CLS
    GOT0230
480 NEXT
490 PRINT"FILE CLOSED -- "'INPUT"TO SEE MENU, HIT (ENTER)";X:CLS:GOTO230
510 CLS:FORI=1TOP1:PRINTNe(I):PRINTRe(I):PRINTAe(I):PRINTPe(I):PRINT:NEXT
520 PRINT: INPUT"TO RETURN TO MENU, HIT (ENTER)";X:CLS:GOTO230
610 CLS:PRINT"ENTER THE NAME FOR THE LINE YOU WISH TO CHANGE (NO COMMAS)"
620 INPUTNS
630 FORI=1TOP1:IFN#=N#(I)G0T0670
640 NEXT
650 PRINT:PRINT"NAME NOT IN FILE -- ":GOT0690
660 PRINT
   PRINT"ENTER THE CORRECTED INFO. : NAME, ADDRESS, PHONE --"
670 INPUTN#(I),R#(I),R#(I),P#(I)
680 PRINT: PRINT"THE FILE NOW READS: "
   :PRINTNO(I):PRINTRO(I):PRINTAO(I):PRINTPO(I):PRINT
690 INPUT"FOR ANOTHER CORRECTION, TYPE 1, OTHERWISE TYPE 0";X
  · IFX=1THEN610ELSECLS:GOTO230
710 CLS:
   INPUT"PREPARE CASSETTE + PLAYER, WHEN READY HIT <ENTER>";X
```

Mailist continued.

#### **TRS-80**

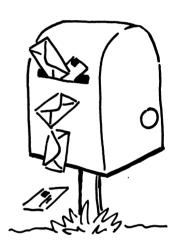
```
720 PRINT: PRINT"COPYING..."
730 PRINT #-1,P1
740 FORI=ITOP1:PRINT #-1,N#(I),R#(I),A#(I),P#(I):NEXT
750 PRINT PRINT COMPLETE -- NOTE TAPE LOCATION PLEASE -- "
    PRINT
760 INPUT"TO RETURN TO MAIN MENU, HIT (ENTER)";X
    :CLS:GOT0230
810 CLS:
    INPUT"PREPARE CASSETTE + RECORDER, WHEN READY HIT <ENTER>";X
                                                                     PRINT
820 PRINT"INPUTING...'
830 INPUT#-1,P1
840 FORI=1TOP1:INPUT#-1,N#(I),R#(I),R#(I),P#(I):NEXT:PRINT
850 INPUT "INPUTING OF DATA COMPLETE. TO SEE MENU PRESS (ENTER)";
                                                                     X:CLS:GOT023
910 CLS:PRINT"FILE VERIFICATION: ":PRINT:PRINT
920 INPUT "NAME FOR SEARCH" JN# PRINT
930 FORI=1TOP1:IFN==N=(I)THENPRINT"FILE FOUND -- ":GOTO960
940 NEXT
950 PRINT"FILE NOT FOUND -- ":INPUT"PRESS (ENTER) TO CONTINUE";
     X:CLS:G0T0230
960 PRINT:PRINTN#(I):PRINTR#(I):PRINTA#(I):PRINTP#(I)
         :INPUT"DO YOU WANT THE NAME PRINTED (Y/N)";X#
         : PRINT
         :IFX#="Y"THENLPRINTN#(I):LPRINTCHR#(24):LPRINTR#(I)
         :LPRINTCHR#(24):LPRINTR#(I):LPRINTCHR#(24)
         :LPRINTP$(I)
970 INPUT"PRESS 1 TO CONTINUE, 0 TO RETURN TO MENU";X
     : IFX=1THEN900ELSECLS: GOTO230
1005 CLS
1010 CLS:PRINT"PRINTING...":PRINT
1020 FORI=1TOP1:LPRINTN#(I):LPRINTR#(I):LPRINTR#(I):LPRINTP#(I):LPRINTCHR#(194):
NEXT
1030 PRINT: PRINT" PRINTING COMPLETE. ": PRINT
1050 INPUT"TO SEE MENU HIT (ENTER)";X
     :CLS:G0T0230
1110 CLS:PRINTTAB(30); "HELP":PRINT:PRINT
1120 PRINT"CREATE ADDRESS FILE(S):
THIS PROCEDURE ALLOWS THE USER TO CREATE THE FILES TO BE USED
IN STORING THE NAMES, ADDRESSES, AND PHONE NUMBERS."
1130 PRINT
1140 PRINT"VIEW ADDRESS FILE(S):
THIS PROCEDURE ALLOWS THE USER TO SEE THE FILES EITHER JUST
CREATED, OR JUST TRANSFERED INTO MEMORY FROM TAPE."
1150 PRINT: INPUT"TO CONTINUE, PRESS 1, ELSE PRESS 0 TO RETURN
TO MAIN MENU";X
          : IFX=0THENCLS: GOTO230
          : IFX=1THENGOTO1160
1160 CLS:PRINT"UPDATE ADDRESS FILE(S):
THIS PROCEDURE ALLOWS THE USER TO CHANGE THE FILES JUST CREA-
TED (FOR TYPO'S), OR FROM TAPE, TO MAKE CHANGES TO PHONE NUM-
BERS OR ADDRESSE(S) AS NEEDED."
1170 PRINT
1180 PRINT"SAVE ADDRESS FILE(S):
THIS PROCEDURE ALLOWS THE USER TO SAVE THE FILE(S) JUST CREATED
OR THE CHANGES JUST MADE."
```

D

**TRS-80** 

Mailist continued

**1190 PRINT** 1200 PRINT: INPUT"TO CONTINUE, PRESS 1, ELSE PRESS 0 TO RETURN TO MAIN MENU" ) X : IFX=0THENCLS:GOTO230 : IFX=1THEN1210 1210 CLS:PRINT"CALL ADDRESS FILE(S): THIS PROCEDURE ALLOWS THE USER TO CALL THE ADDRESS FILE(S) FROM CASSETTE-TAPE DATA FILES." 1220 PRINT 1230 PRINT"VERIFY ADDRESS FILE(S): THIS PROCEDURE ALLOWS THE USER TO VERIFY THE ADDRESS FILE(S). THIS IS USED WHEN THE USER WISHES TO SEARCH FOR A PARTICULAR NAME, ADDRESS AND PHONE NUMBER." 1240 PRINT: INPUT"TO CONTINUE, PRESS 1, ELSE PRESS 0 TO RETURN TO MAIN MENU";X : IFX=0THENCLS:GOT0230 : IFX=1THENGOT01250 1250 CLS:PRINT"PRINT ADDRESS FILE(S): THIS FUNCTION ALLOWS THE USER TO GET A HARD COPY OF THE FILES EITHER ON PRINTER PAPER, OR STICKY LABELS FOR ENVELOPES." 1260 PRINT 1270 PRINT"HELP: THIS FUNCTION." 1280 PRINT 1290 PRINT"QUIT: END THE PROGRAM." 1300 PRINT 1310 INPUT"TO RETURN TO MENU, PRESS (ENTER>";X : CLS :G0T0230 9999 CLS: :POKE16396,201 : END



# Australia's Top Motorcycle Monthly



#### **TRS-80**

#### MATH LAUNCH

Math Launch is an educational game featuring high-res graphics, music and sound. The aim of the game is to build a rocket and launch it by answering multiplication questions. There is also a choice of three skill levels.

John Pospisil Bulli NSW

```
1880 SOUND2;4,5;4,14;4
85
     CLS
                                       1890 SOUND2;3,5;3,14;3
7
     CHAR63,0000003C003C
                                       1900 SOUND2;1,5;1,14;1
10
     G0T01000
506 NEXT
                                       1910 SOUND2;4,5;4,14;4
1000 GOSUB9000
                                      1911 SOUND2;3,9;3,17;3
                                      1912 SOUND2;1,7;1,16;1
1010 GOSUB9800
                                      1913 SOUND2;3,7;3,16;3
1914 SOUND2;1,5;1,14;1
1015 CLS
1020 PRINT"WHICH SKILL LEVEL?"
                                      1915 SOUND2;3,5;3,14;3
1030 PRINT
1040 PRINT"
             (A) VERY EASY"
                                      1916 SOUND2;1,4;1,12;1
1050 PRINT
                                       1917 SOUND2;7,5;7,14;7
1060 PRINT"
             (B) INTERMEDIATE"
                                       1920 FORN=13T023
                                       1930 FORM=25T022
1070 PRINT
1080 PRINT" (C) TOUGH"
                                       1950 PLOTM, N, 32
                                       1960 NEXT
1085 FORN=1T010
                                       1965 SOUNDN;0
1086 PRINT
                                       1970 NEXT
1087 NEXT
                                      1975 U=0
1090 IFPEEK(29)=1THEN1100
                                      1980 POKE219,208
1095 GOTO1090
                                   1985 POKE218,228
1986 PRINT'AH WELL,LETS START AGAIN"
1100 IFPEEK(25)=17THENA=7
1110 IFPEEK(25)=6THENA=15
                                     1987 FORN=1T0300
1120 IFPEEK(18)=72THENA=40
1130 FORN=1T020
                                      1988 NEXT
                                       1990 GOSUB5000
1140 PRINT
1150 NEXT
                                       1995 GOTO1800
1160 IFA=0THEN1020
                                      2000 REM
                                      2005 IFU=11THEN4100
1500 RFM SRFFN
1501 CHAR180,FFC3A59999A5C3FF
                                    2010 GOSUB3000+U*100
                                       2020 GOSUB5000
1502 FORN=22T023
                                      2030 11=11+1
1503 FORM=14T024
1504 PLOTN, M, 180
                                      2040 GOTO1800
                                       2500 FORN=1T0256
1505 NEXT
1507 NEXT
                                       2510 POKE4098,N
                                       2520 FORM=1T020
1510 FORN=25T027
1520 PLOTN, 24, 200
                                      2530 NEXT
1530 NEXT
                                       2540 NEXT
1540 FORN=18T020
                                       3000 PLOT25,23,96
1545 PLOTN, 24, 252
                                       3010 PLOT26,23,97
                                      3020 PLOT27,23,98
1546 NEXT
1550 PLOT19,22,250
                                      3030 RETURN
1555 PLOT19,23,251
                                      3100 PLOT25,22,104
                                       3110 PLOT26, 22, 105
1580 POKE219,208
                                      3120 PLOT27,22,106
1590 POKE218,228
1600 PRINT"LET'S START BUILDING " 3130 RETURN
                                      3200 PLOT25,21,107
1610 FORN=1T0250
1620 NEXT
                                       3210 PLOT26,21,102
1630 POKE218,228
                                       3220 PLOT27, 21, 107
1640 PRINT"
                                      3230 RETURN
1650 U=0
                                       3300 PLOT25, 20, 108
1800 REM QUESTION/ANSWER
                                      3310 PLOT26,20,109
                                      3320 PLOT27,20,108
1805 IFU=11THENGOSUB4300
1810 B=RND(A)
                                      3330 RETURN
1820 C=RND(A+5)
                                     3400 PLOT25,19,108
                                     3410 PLOT26,19,110
1830 POKE219,208
1840 POKE218,230
                                      3420 PLOT27, 19, 108
                                     3430 RETURN
1850 PRINTB;" ";C;
                                     3500 PLOT25,18,113
1861 INPUTT$
1862 D=UAL(T$)
                                      3510 PLOT26, 18, 112
1870 IFD=C*BTHEN2000
                                       3520 PLOT27, 18, 113
```

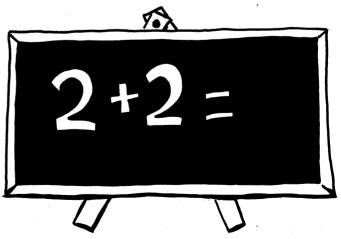
#### **TRS-80**

```
3530 RETURN
3600 PLOT25,12,113 .
3610 PLOT26,17,114
3620 PL0T27,17,113
3630 RETURN
3700 PLOT25,16,111
3710 PLOT27, 16, 111
3715 PLOT26, 16, 111
3720 RETURN
3800 PLOT25, 15, 120
3810 PLOT26, 15, 108
3820 PLOT27, 15, 121
3830 RETURN
3900 PLOT25, 14, 122
3910 PLOT26, 14, 124
3920 PLOT27,14,123
3930 RETURN
3999 FND
4000 PLOT25,13,128
4010 PLOT26, 13, 129
4020 PLOT27,13,130
4030 RETURN
4100 FORN=18TO20
4110 PLOTN, 24, 32
4111 NEXT
4119 PLOT19, 23, 32
4120 PLOT19,22,32
4125 FORM=13T024
4126 PLOT22, M, 32
4127 PLOT23, M, 32
4128 NEXTM
4130 GOSUB5000
4135 COLOR26,7,5
4140 CHAR200, FF7F7F7762
4145 POKE219,210
4150 POKE218,226
4155 POKE4098,0
4160 FORN=1T024
4170 PRINT
4180 NEXT
4190 POKE219,209
4200 POKE218,68
4210 CHAR2, C0A0904F300C03
4220 CHAR3,000000F01010F0
4230 CHAR4,030509F20C30C0
4240 CHAR5,0000000F08080F
4250 PRINT" WELL DONE"
4255 FORN=2T010
4256 POKE4098,N
4260 FORM=1T020
4270 NEXT
4280 NEXT
4290 SOUND0;0
4295 GOTO6000
4299 END
4300 FORN=1T020
4305 POKE218,100
4310 PRINT"DATA FOR LAUNCH"
```

4320 POKE218,100

4330 PRINT" 4340 NEXT 4350 RETURN 5000 POKE219,208 5010 POKE218,228 5020 PRINT" 5030 RETURN 6000 SOUND26;3 6005 SOUND21;2 6010 SOUND21;1 6015 SOUND23;3 6020 SOUND21;3 6025 SOUND1;3 6030 SOUND9;3,19;3,25;3 6040 SOUND2;3,18;3,26;3 6050 FORN=1T0200 6060 NEXT 6070 PRINT"PRESS ANY KEY FOR NEW GAME" 6080 FORN=1T010 6090 PRINT 6100 NEXT 6110 IFPEEK(29)=1THENRUN 6120 GOT06110 8000 FORN=1T0255 8010 POKE4098,N 8020 FORM=1T050 8030 NEXT 8040 NEXT 8999 FND 9000 CHAR96, FF0F0F1F1F3F3F7F 9001 CHAR97,00FF5AA5FF 9002 CHAR250,01010103030E1E3E 9003 CHAR251, 3E76F606060F0F0F 9004 CHAR1,0081422418244281 9005 CHAR252, FFFFFFFFFFFFFFF 9010 CHAR92, FFFFFFFFFFFFFF 9020 CHAR98, FFF0F0F8F8FCFCFE 9030 CHAR104,FF0303030303

9040 CHAR105 FEC303030303 9050 CHAR106, FFC0C0C0C0C0 9060 CHAR102,000000FFFFFFFFF 9070 CHAR108,00 9080 CHAR109,00003C3C18181818 9090 CHAR110,1800001C20180438 9100 CHAR112,2424000024242424 9110 CHAR113,00 9120 CHAR114,000000000018243C 9125 CHAR111, FØFØFØFØFØFØFØF 9130 CHAR120,0F0F1F1F3F3F7FFF 9140 CHAR121, FØFØF8F8FCFCFEFF 9150 CHAR122,0F0F0F0F0F0F0F0F 9160 CHAR123, F0F0F0F0F0F0F0F0 9170 CHAR124, FFC3C3C3FFFFFFFF 9180 CHAR128,00000000003030F0F 9195 CHAR129,003C7EFFFFFFFF 9200 CHAR130,000000000C0C0F0F0 9205 CHAR200,0000FFA55AFF 9210 RETURN 9800 COLOR0,13 9810 COLOR13,2,5 9820 COLOR14,2,16 9830 COLOR15,2,15 9835 COLOR16,16,5 9840 COLOR17,2,5 9845 COLOR26,2,5 9846 COLOR32,11,5 9850 FORN=6T012 9860 COLORN, 2, 5 9870 NEXT 9877 COLOR1,2,5 9880 COLOR5,2,5 9885 COLOR23,2,5 9900 RETURN 9997 POKE4098,208 9998 POKE4098,0 9999 POKE4098,207



# Apple II 📹

#### CRICKET

In this game you field while the computer bats. The ball is hit from a random end in a random direction and you move around the screen, using the keyboard, to stop the ball.

If you find the game too easy, change the values in lines 165-190 from 2 and -2 to 3 and -3 (or even 4). Don't change the D values in line 170

Type carefully because mistakes will send the program to the error handling routine and it will be hard to find the error. If you think you have an error, use the TRACE command and run the program. NO TRACE turns the TRACE off.

Good luck and have fun. Richard de Meester Airport West, Vic

```
PRESS ANY KEY TO START
5 ONERR GOTO 1100
                                                  ";: GET B$
6 TEXT : HOME
10 LS = "CRICKET?!"
                                            95 DIM BX(200): DIM BY(40)
    HTAB 12: VTAB 15: SPEED= 240:
FOR Z = 1 TO 9:Z$ = MID$ (
                                            98 \text{ TSC} = 0:PSC = 0
                                            100 GOSUB 1000: GOSUB 1020
                                            110 GOSUB 800
     L$,Z,1): FOR ZZ = 32 TO 255:
     PRINT CHR$ (ZZ);: IF CHR$ (ZZ) < > Z$ THEN CALL - 1
                                            120 R = 2: GOSUB 1050
                                            130 E = X:R = 7: GOSUB 1050:D = X
      008: NEXT
20 X = PEEK ( - 16336) - PEEK ( - 16336) - X = PEEK ( - 16336) - X =
                                             140
                                                  GOSUB 800
                                                  IF E = 1 THEN E = 125
                                                  IF E = 2 THEN E = 155
                                            160
     PEEK ( - 16336) - PEEK ( - 16336) - X = PEEK ( - 16336) - PEEK ( - 16336): NEXT Z
                                                  IF D = 1 THEN DX =
                                             165
                                             170
                                                  IF D = 2 THEN DX = 0:DY =
      : SPEED= 255: PRINT
    FOR I = I TO 500: NEXT : HOME
                                                  IF D = 3 THEN DX = 2:DY =
                                             175
     :L$ = "BY RICHARD DE MEESTER
      , AIRPORT WEST."
                                                  TFD = 4 THEN DX = 2:DY = 2
                                             180
     = 1: GOSUB 30: GOTO 55
                                                  IF D = 5 THEN DX = 0:DY = 2
                                             185
    POKE 34, V:ZZ = LEN (L$) / 2:
30
                                                  IF D = 6 THEN DX =
                                                                        - 2:DY =
                                             190
      SPEED= 160: VTAB 23
    FOR Z = 1 TO LEN (L$) / 2: HTAB
1: PRINT MID$ (L$,ZZ + 1 -
31
                                                  TFD = 7 AND E = 125 THEN DY
                                             195
                                                   = 0:DX = 4
     Z,1);: HTAB 40: PRINT MIDS
                                                  IF D = 7 AND E = 155 THEN DY
                                             200
     (L\$,ZZ + Z,1);: NEXT
                                                   = 0:DX =
                                                              - 4
35
    SPEED= 65: FOR Z = 1 TO 23 -
                                             205
                                                  GOSUB 800
     V - ZZ: PRINT : NEXT : SPEED=
                                                  IF XP > 50 AND XP < 200 AND
                                             206
                                                  YP < 120 AND YP > 10 THEN HOME
: VTAB 23: PRINT "FURTHER BA
     255
40
    FOR Z = 1 TO ZZ: VTAB V: HTAB
     1: PRINT MID$ (L$,ZZ + 1 - Z,Z);: HTAB 41 - Z: PRINT MID$
                                                  CK, PLEASE!": GOTO 205
                                                  HOME : VTAB 23: PRINT "COMPU
     (L$,ZZ + 1,Z);: VTAB 24: PRINT
                                                  TER SCORE: ";TSC;" YOUR SC
      : NEXT
                                                  ORE: ";PSC
    FOR Z = 1 TO 20 - ZZ: VTAB V:
                                             215 BX(1) = E + DX:BY(1) = 80 + D
      HTAB Z: PRINT " "; LEFTS (L
      $, ZZ);: HTAB 41 - ZZ - Z: PRINT
                                             220
                                                  HPLOT E, 80 TO BX(1), BY(1)
      RIGHTS (L$,ZZ);" ":: NEXT
                                             222 G = PEEK ( - 16336) * PEEK
    SPEED= 255: POKE 34,0: RETURN
                                                  ( - 16336)
                                             225 I = 2
55 V = 3:L$ = "NO COPYRIGHT < C > O
                                             230
                                                  GOSUB 800:J = I - 1:BY(I) =
     N THIS GAME!": GOSUB 30
                                                   BY(J) + DY:BX(I) = BX(J) + D
    PRINT : PRINT : PRINT "
     OU ARE FIELDING AT A GAME OF
                                                  HPLOT BX(J), BY(J) TO BX(I), B
                                             235
           ": PRINT "CRICKET. YO
                                                   Y(T)
     U ARE A BOX AND YOU HAVE TO
                                                  IF XP = \langle BX(I) AND XP + 29 \rangle
                                             240
      ": PRINT "MOVE IT AROUND AND
                                                   \Rightarrow = BX(I) AND YP = < BY(I
STOP THE BALL.
65 PRINT " KEYS
                                                   ) AND YP + 29 >
                                                                    = BY(I) THEN
             KEYS ARE: U I
                                                  242
     O": PRINT : PRINT : PRINT "
                                             241 I = I + 1: GOTO 230
                            L": PRINT
                                             242 IF I < 15 THEN PSC = PSC + 4
     : PRINT : PRINT "
                                                   : GOTO 245
          M . ": PRINT : PRINT
                                                  IF I < 25 THEN PSC = PSC + 3
     · PRINT
                                                   :TSC = TSC + 1: GOTO 245
                THE FIRST ONE TO
                                             244 \text{ PSC} = \text{PSC} + 1:\text{TSC} = \text{TSC} + 2
    PRINT "
     100 WINS": VTAB 24: PRINT "
                                                  IF TSC > = 100 OR PSC >
         PRESS ANY KEY";: GET B$
                                                   100 THEN 300
    HOME : PRINT "
                        THE EARLIE
                                                  HOME : VTAB 23: PRINT "COMPU
     R YOU STOP THE BALL THE ": PRINT
                                                   TER SCORE: ";TSC;"
                                                                       YOUR SC
     "HIGHER SCORE YOU WILL GET F
                                                   ORE: "; PSC
     OR THE STOP. "
                                                  GOTO 100
78 PRINT "THE BALL WILL BE HIT F
                                                  TEXT : HOME : IF PSC > TSC THEN
     ROM A RANDOM END ": PRINT "
                                                   PRINT "YOU WON, "; PSC; " TO
     IN A RANDOM DIRECTION.": PRINT
                                                   "; TSC; "!"
     : PRINT : PRINT
                                             305
                                                  IF TSC > PSC THEN PRINT "TH
                 IF YOU ARE TOO CL
                                                   EY BEAT YOU, ":TSC: " TO ":PS
    PRINT
     OSE YOU WILL BE ": PRINT "
                                                   C;"!
     ASKED TO STAND FURTHER BACK.
                                                  IF TSC = PSC THEN PRINT "IT
                                             310
                                                   WAS A DRAW, ";TSC;" ALL!
      ". PRINT . PRINT : PRINT : PRINT
```

# Apple II

```
1020 HPLOT XP.YP TO XP + 29.YP TO
315 PRINT : PRINT : PRINT "AGAIN
                                              YD = 10: GOTO 920
      <Y/N>?";: GET AS: IF AS = "
I" THEN PRINT : PRINT : PRINT
                                         890 IF MS = "," THEN XD = 0:YD =
                                                                                         XP + 29, YP + 29 TO XP, YP + 2
                                                                                         9 TO XP, YP: RETURN
                                              10: GOTO 920
                                                                                    1050 X = INT (RND (1) * R) + 1
     "BYE!": END
                                         900 IF MS = "." THEN XD = 10:YD =
     IF AS = "Y"
                                                                                    1060
                                                                                          RETURN
320
                 THEN 98
                                              10: GOTO 920
                                                                                    1100
     GOTO 315
325
                                         910 RETURN
     POKE 34,V:ZZ =
                      LEN (L$) / 2
                                                                                          VTAB 21: PRINT "FFFF OO U
330
                                         920 HCOLOR= 0: GOSUB 1020
     : SPEED= 160: VTAB 23
                                                                                           U RRR
                                         930 \text{ XP} = \text{XP} + .\text{XD}: \text{YP} = \text{YP} + \text{YD}
     IF PEEK ( - 16384) > 127 THEN
                                        931 IF XP < 0 THEN XP = 0
932 IF XP > 250 THEN XP =
                                                                                          PRINT "F
                                                                                    1105
                                                                                                         OUURR
800
     820
                                              IF XP \rightarrow 250 THEN XP = 250
                                                                                    1110
                                                                                          PRINT "FFF O
                                                                                                          O U U RRR"
     RETURN
                                         933 IF YP > 130 THEN YP = 130
934 IF YP < 0 THEN YP = 0
810
                                                                                          PRINT "F
                                                                                    1115
                                                                                                        OO UU R R
820
     GET MS
     IF M$ = "U" THEN XD = - 10:
830
                                         940 HCOLOR= 7: GOSUB 1020
                                                                                    1116 FOR F = 1 TO 250:G = PEEK
     YD = -10: GOTO 920
                                         945 G = PEEK ( - 16336) + PEEK
     IF M$ = "I" THEN XD = 0:YD =
                                                                                         ( - 16336): NEXT
840
                                              (-16336)
                                                                                    1117 TSC = TSC + 4: IF TSC = > 1
      - 10: GOTO 920
     IF MS = "O" THEN XD = 10:YD =
                                         950 IF XP < 170 AND XP > 80 AND
                                                                                         00 THEN 300
                                                                                   1118 HOME : VTAB 23: PRINT "COMP
                                              YP < 100 AND YP > 30 THEN GOSUB
       - 10: GOTO 920
     IF M$ = "J" THEN XD = - 10:
                                                                                                                 your s
                                              1010
                                                                                         UTER SCORE: ";TSC;"
860
                                                                                         CORE: "; PSC
                                         960 RETURN
     YD = 0: GOTO 920
                                        1000 HGR: HCOLOR= 7
1010 HPLOT 120,75 TO 120,84 TO 1
                                                                                    1120 GOTO 100
870
     IF M$ = "L" THEN XD = 10:YD =
     0: GOTO 920
                                                                                    1244 PSC = PSC + 2:TSC = TSC + 2
     IF MS = "M" THEN XD = -10:
                                              59,84 TO 159,75 TO 120,75: RETURN JPR#0
```

# THIME

# ERICSSON **S**



Complete compatibility Complete · software Complete 3 year warranty Complete training programme Complete-financing Complete customer hot line Complete accessories



602 Maroondah Highway, Mitcham, Victoria 3132. 2 (03) 873 1122



plus \$2.00 postage & packaging

EDD runs on Apple II, II+ (including most compatibles), Ilc, Ile and III (in emulation mode), 3.3 disk drives

#### ESSENTIAL DATA DUPLICATOR IIITM

- EDD rarely needs parameter changing.
- Automatically finds the beginning of each track.
- Unlike any of the Copycards; EDD backs up the entire disk - not just what is in memory
- · Accurately finds "auto-sync" bytes and their lengths.
- Can copy 1/4 and 3/4 tracks.

EDD is the most powerful disk duplicator available for your Apple<sup>TM</sup> computer. Unlike the Copycards, which only copy single load programs, EDD backs up your entire disk. EDD can back up more protected software than all other copy

programs or Copycards put together. Since EDD is automatic, you will no

longer have to change parameters to duplicate most disks, although every

parameter is fully documented in our extensive manual. We also provide updated EDD program lists

Order by phone

To order your copy send Cheque or Money Order





UTILICO SOFTWARE 83 Hall Street, Bondi Beach, NSW 2026 Telephone: (02) 30-2105

# MICTO Dee Australia's home grown computer

